

APPLIED COMPUTING

VCE UNITS 1 & 2

Selina Dennis
Gary Bass
Therese Keane
Anthony Sullivan

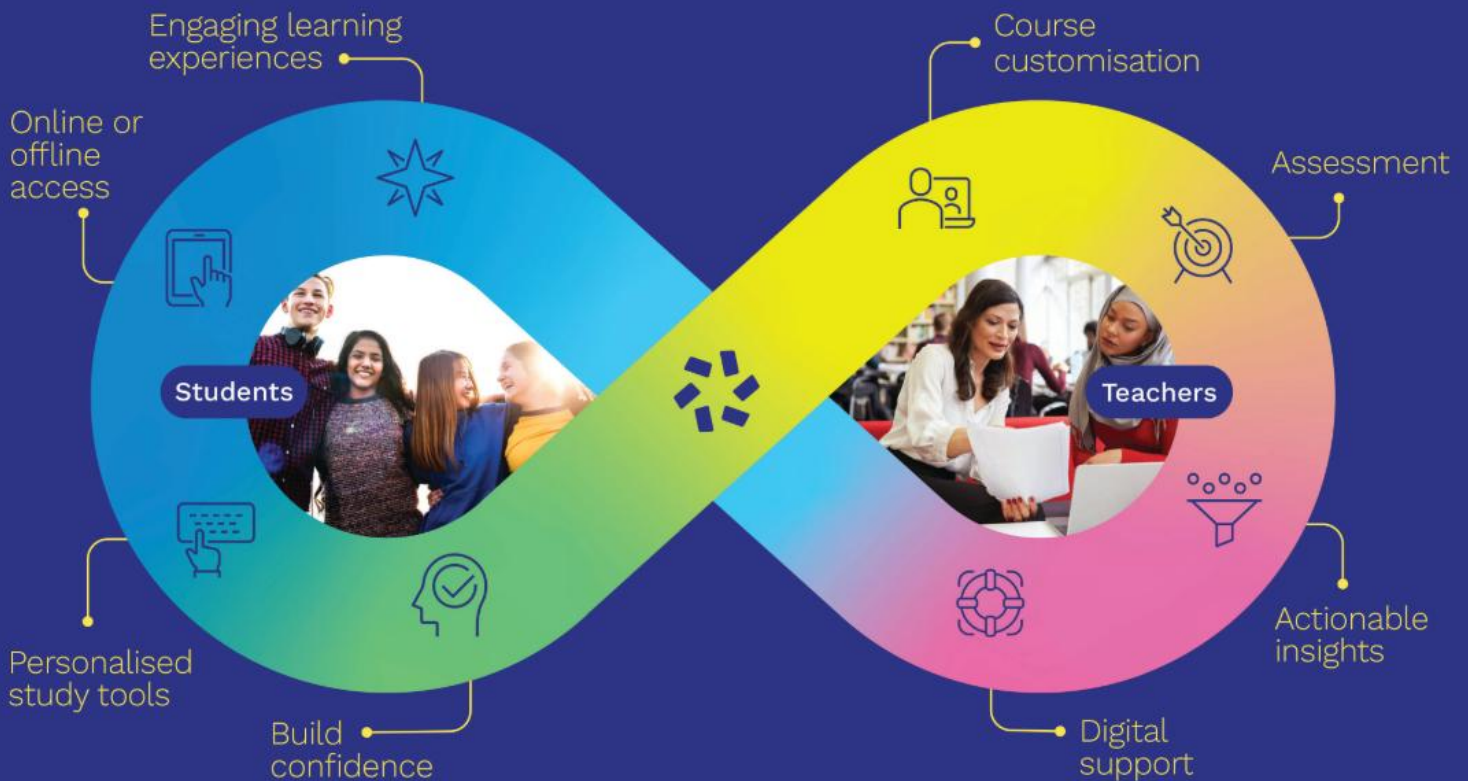
 Nelson



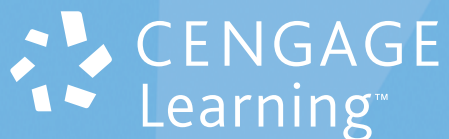
8E

Learning that puts you at the centre

...and on the learning path to success.



Find everything you need to access
your Nelson MindTap course at
cengage.com.au/nelsonmindtap



COPYRIGHT NOTICE

Copyright in this work is owned by Cengage Learning Australia ("the work"). A condition of purchase of this electronic version of the work is that you agree to respect the copyright in the work, abide by the Copyright Act 1968 and specifically agree not to transfer, sell, assign, misuse, copy or transmit an electronic or other version of the work to any third party.

Please note: This product is accompanied by a licence (single user, network or adoption) governing the terms and conditions of its use.

This is a legal agreement between the you, (the “Customer”) and Cengage Learning Australia Pty Limited (ABN 14 058 280 149) (the “Licensor”) which provides the terms and conditions of this non-exclusive licence and the limited warranty for the Product. Use of the Product indicates an acknowledgement that the Customer has read and agreed to be bound by the terms and conditions of this Agreement. If you do not agree to these terms and conditions, return the Product to the place of purchase within 15 days of the date of purchase (with proof of purchase) for a full refund

1. Licence Grant

You do not receive title to the Product. Copyright in the Product (which includes all images, photographs, video, animations, audio, music and text incorporated in the Product, including all of the accompanying printed material) is owned by the Licensor and/or its suppliers and is protected by Australian copyright laws. The Licensor grants you a non-exclusive licence to use the Product subject to the restrictions and terms set out in this Agreement.

2. A Licence allows you to:

Use the Product on your computer. The Customer represents that they shall in no way place the Product in the public domain or in any way compromise our copyright in the Material. You agree to take reasonable steps to protect our copyright.

3. You may not:

Alter, modify, translate, reverse engineer, decompile, or adapt the software or create derivative works based on the Product. Make further copies by any means technological, electronic, digital whatsoever without the written permission of the Licensor. Rent or transfer all or any part of your rights under this Agreement. Remove or alter any copyright or other proprietary notice or label attached to the software.

4. Termination

Any failure to comply with the terms and conditions of this agreement will result in the automatic termination of this licence. Upon termination of this licence for any reason, the Customer must destroy or return to the Licensor all copies of the software and accompanying documentation.

5. Warranties

To the extent permitted by law, the Licensor’s liability for any breach of the warranty or any term implied by law into this licence is limited to the lowest cost of replacing the goods, acquiring equivalent goods or having the goods repaired.

APPLIED COMPUTING

VCE UNITS 1 & 2

RDVector / Adobe Stock Photos

Selina Dennis
Gary Bass
Therese Keane
Anthony Sullivan

8E

Applied Computing VCE Units 1&2

8th Edition

Selina Dennis

Gary Bass

Therese Keane

Anthony Sullivan

ISBN 9780170483889

Product manager: Melissa O'Shea

Content developer: Morgan Beggs

Content manager: Alice Kane/Katie McCappin

Associate Content project manager: Ananya Sarkar

Text designer: Leigh Ashforth, Watershed Art & Design

Cover designer: Cengage Creative Studio

Editor: Stephen Roche

Proofreader: Sylvia Marson

Permissions/Photo researcher: Lumina Datamatics

Cover: stock.adobe.com/RDVector; stock.adobe.com/iiierlok_xolms

Typeset by Straive

Any URLs contained in this publication were checked for currency during the production process. Note, however, that the publisher cannot vouch for the ongoing currency of URLs.

Extracts from the VCE Applied Computing Study Design (2025–2028) are reproduced by permission, © VCAA. VCE® is a registered trademark of the VCAA. The VCAA does not endorse or make any warranties regarding this study resource. Current VCE Study Designs, past VCE exams and related content can be accessed directly at www.vcaa.vic.edu.au

© 2025 Cengage Learning Australia Pty Limited

Copyright Notice

This Work is copyright. No part of this Work may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without prior written permission of the Publisher. Except as permitted under the *Copyright Act 1968*, for example any fair dealing for the purposes of private study, research, criticism or review, subject to certain limitations. These limitations include: Restricting the copying to a maximum of one chapter or 10% of this book, whichever is greater; providing an appropriate notice and warning with the copies of the Work disseminated; taking all reasonable steps to limit access to these copies to people authorised to receive these copies; ensuring you hold the appropriate Licences issued by the Copyright Agency Limited ("CAL"), supply a remuneration notice to CAL and pay any required fees. For details of CAL licences and remuneration notices please contact CAL at Level 11, 66 Goulburn Street, Sydney NSW 2000, Tel: (02) 9394 7600, Fax: (02) 9394 7601
Email: info@copyright.com.au
Website: www.copyright.com.au

For product information and technology assistance,
in Australia call **1300 790 853**;
in New Zealand call **0800 449 725**

For permission to use material from this text or product, please email aust.permissions@cengage.com

National Library of Australia Cataloguing-in-Publication Data

A catalogue record for this book is available from the National Library of Australia

Cengage Learning Australia

Level 7, 80 Dorcas Street
South Melbourne, Victoria Australia 3205

For learning solutions, visit cengage.com.au

Printed in China by 1010 Printing International Limited.
1 2 3 4 5 6 7 27 26 25 24 23



Contents

Preface	v
About the authors	vi
How to use this book	vii
Outcomes	x
Problem-solving methodology	xiii

Unit 1

Introduction	1	Chapter 3 Designing software	108
Chapter 1 Data analysis	2	Programming and scripting languages	109
Understanding research	3	Software development tools	110
Data and information	5	Emerging trends in programming and AI	112
Primary and secondary data and information	5	Characteristics of a software solution	113
Referencing primary sources	10	Determining scope	113
Interpretation of information for communication and decision-making	12	Functional and non-functional requirements	114
Quality of data and information	12	Constraints	117
Data types relevant to selected software tools	15	Characteristics of data types	119
Privacy	18	Types of data structures	122
Australian Privacy Principles	19	Representing designs	123
Security to protect personal information	22	Chapter 4 Developing software	133
Ethical issues	25	Processing features of a programming language	134
Chapter 2 Data visualisations	39	Object-oriented programming	145
Structural characteristics of database and spreadsheet software	40	Programming naming conventions	148
Software functions	46	Internal documentation	149
Structured Query Language	48	Validation techniques	151
Data visualisations	61	Testing and debugging techniques	152
Types of data visualisations	65	Preparing for Unit 1, Outcome 2	168
Design tools	80	Unit 1 Exam practice	169
Formats and conventions	85		
Preparing for Unit 1, Outcome 1	107		

Unit 2

Introduction	173	Chapter 7 Network security	259
Chapter 5 Innovative solutions	174	Key network hardware and software components	260
Inspiring and driving innovation	175	Networking hardware and software	264
Functions and capabilities of digital systems	178	Defending networks	267
Emerging technologies	189	Wired, wireless and mobile communications technology	268
Impact of emerging technologies	194	Technical underpinnings of networks	276
Chapter 6 Development and evaluation of an innovative solution	205	Network diagrams	279
Key legislation and emerging technologies	206	Artificial intelligence and network security	280
Documenting and monitoring progress	217	Risks of using networks in a global environment	281
Collecting data	221	Reducing risks and mitigating threats	282
Characteristics of creative and innovative solutions	222	Cryptographic techniques to protect networks	283
Solution specifications	223	Security threats on networks	287
Generating possible design ideas	229	Preventative practices to reduce risks and mitigate threats	292
Developing the preferred design	233	Ethical hacking	294
Techniques for testing and evaluating innovative solutions	238	Key legislation	295
Preparing for Unit 2, Outcome 1	258	Emerging trends in cyber security	297
		Preparing for Unit 2, Outcome 2	307
		Unit 2 Exam practice	308
		Index	312

Preface

This eighth edition of *Applied Computing VCE Units 1 & 2* incorporates the changes to the VCAA VCE Applied Computing Study Design that took effect from 2025.

This textbook looks at how individuals and organisations use, and can be affected by, information systems in their daily lives.

We believe that teachers and students require a text that focuses on the **areas of study** specified in the **study design**, and that presents information in a sequence that allows easy transition from theory into practical assessment tasks. We have therefore written this textbook so that a class can begin at Chapter 1 and work their way systematically through to the end. Students will encounter material relating to the **key knowledge** dot points for each **outcome** before they reach the special section that describes the outcome. The study design outlines key skills that indicate how the knowledge can be applied to produce a solution to an information problem. These outcome preparation sections occur regularly throughout the textbook, and flag an appropriate point in the student's development for each outcome to be completed. The authors have covered all key knowledge dot points for the outcomes from the Applied Computing VCE Units 1 & 2 course.

Our approach has been to focus on the key knowledge required for each school-assessed outcome, and to ensure that students are well prepared for these; however, there is considerable duplication in the study design relating to the knowledge required for many of the outcomes. We have found that, with an outcomes approach, we are sometimes covering the same material several times. For example, knowledge of requirements and scope is listed as key knowledge for more than one outcome. In these cases, we have tried to provide a general coverage in the first instance, and specifically apply the concept to a situation relevant to the related outcome on subsequent encounters.

The authors assume teachers will develop the required key skills with their students within the context of the key knowledge addressed in this textbook and the resources available to them.

We have incorporated a margin column in the text that provides additional information and reinforcement of key concepts. The margin column also includes activities related to the topics covered in the text, and consideration of issues relevant to the use of information systems.

Outcome features are included at several points in the book, indicating the nature of the tasks that students are to undertake in the completion of the school-assessed outcome. The steps required to complete the outcome are listed, together with advice and suggestions for approaching the task. The output and support material needed for submission are described. Sample tasks and further advice relating to the outcomes are available on Nelson MindTap.

The chapters are organised to present the optimum amount of information in the most effective manner. The text is presented in concise, clearly identified sections to guide students through the text. Each chapter is organised into the sections described on pages vii–viii.

About the authors

Selina Dennis is a Software Development and English Language teacher for the Department of Education and has been heavily involved in past and present Computing study designs. Selina has a Bachelor of Arts and Science in Computer Science and Linguistics from the University of Melbourne, and has a particular interest in Computational Linguistics. She spent several years in California in the computing industry as an Engineering Manager and Director of Engineering before entering teaching.

Gary Bass has taught VCE Applied Computing at Year 11 and Year 12 in an online course environment at Virtual School Victoria. Previously he has taught VCE Physics, as well as developed and delivered middle school ICT courses. Gary has presented at DLTV DigiCON and the annual IT teachers conference on many topics, including 'Pop-up Makerspace', 'Big Data requires huge analysis – data visualisation', 'AR + VR = Mixed reality', and 'Marshall McLuhan – Medium is the message'.

How to use this book

KEY KNOWLEDGE

The key knowledge from the VCAA Applied Computing VCE Units 1 & 2 Study Design that you will cover in each chapter is listed on the first page of each chapter. The list includes key knowledge specified in the outcome related to the chapter.

FOR THE STUDENT

The first page of each chapter includes an overview of the chapter's contents so that you are aware of the material you will encounter.

CHAPTERS

The major learning material that you will encounter in the chapter is presented as text, photographs, screenshots and illustrations. The text describes in detail the theory associated with the stated outcomes of the Applied Computing VCE Units 1 & 2 Study Design in easy-to-understand language. The photographs show hardware, software and other objects that have been described in the text. Illustrations are used to demonstrate concepts that are more easily explained in this manner.

Throughout the chapter, glossary terms are highlighted in bold, light-blue text and you can find their definitions at the end of the chapter, in **Essential terms**.

MARGIN COLUMN

The margin column contains further explanations that support the main text, weblink icons, additional material outside the study design and cross-references to material covered elsewhere in the textbook. Issues relevant to Applied Computing that you can discuss with your classmates are also included in the form of 'Think about Applied Computing' boxes (right).

CHAPTER SUMMARY

The chapter summary at the end of each chapter is divided into two main parts to help you review each chapter.

Essential terms lists the glossary terms that have been highlighted throughout the chapter.

Important facts is a list of summaries, ideas, processes and statements relevant to the chapter, in the order in which they occur in the chapter.

1.4 THINK ABOUT APPLIED COMPUTING

In theory, a Boolean data type only requires 1 bit of memory space. Conduct research to find out why it actually needs to take up 1 byte of memory.

TEST YOUR KNOWLEDGE

These are short-answer questions that are provided to help you when reviewing the chapter material. The questions are grouped, and identified with a section of the text, to allow your teacher to direct appropriate questions based on material covered in class. Teachers will be able to access answers to these questions on **Nelson MindTap**.

APPLY YOUR KNOWLEDGE

Each chapter concludes with a set of questions requiring you to demonstrate that you can apply the theory from the chapter to more complex questions. The style of questions reflects what you can expect in the end-of-year examination. Teachers will be able to access suggested responses to these applications on **Nelson MindTap**.

PREPARING FOR THE OUTCOMES

This section appears at points in the course where it is appropriate for you to complete an outcome task. The information provided describes what you need to do in the outcome, the suggested steps to be followed in the completion of the task and the material that needs to be submitted for assessment.

EXAM PRACTICE

Each unit concludes with a set of exam-style questions requiring you to demonstrate that you can apply the theory from the chapter to more complex questions. These questions include multiple-choice questions as well as short-answer questions based around small case studies. Teachers will be able to access answers to these questions on **Nelson MindTap**.

NELSON MINDTAP

This product includes **Nelson MindTap**, an online learning environment that puts you at the centre. Nelson Applied Computing Nelson MindTap gives you access to an eText with additional resources. Nelson MindTap products are not for individual use; they are for adoption, class set or book hire. Nelson MindTap is teacher-led courseware and requires an active course for access. This is a student activation code that can only be used to access a Nelson MindTap course as part of a school's adoption. Please contact your learning consultant to discuss purchasing options.



An online learning space that provides students with tailored learning experiences.

- Access tools and content that make learning simpler yet smarter.
- **Flexible formats:** choose how you navigate using either the online eText, or offline PDFs.
- Margin links in the student book signpost multimedia student resources found on Nelson MindTap.



For students:

- **Access** weblinks associated with the content to build upon your understanding of the concepts through real-world applications.
- **Analyse** additional material such as spreadsheets, charts and infographics to enhance your learning experience.
- **Navigate** your own path, accessing the content and support as you need it.
- **Download** individual chapters as PDFs for easy offline access to the material.

For teachers:

- **Assign** assessment and revision tasks with downloadable end-of-chapter tests and practice exams.

- Get expert and detailed guidance on **preparing SATs** with sample tasks, marking advice and suggested solutions.
- Use **teaching plans** to support easy program and assessment planning.
- Access expert **answers and solutions** to test your knowledge, apply your knowledge, exam-style questions and end-of-chapter tests.
- Help build your students' exam readiness with **Cognero Assess** – a test bank containing hundreds of questions and answers to create, assign or export formative and summative tests.

Complimentary access to these resources is only available to teachers who use this book as part of a class set, book hire or booklist. Contact your Cengage Education Consultant for information about access and conditions.

Security & privacy:

Nelson MindTap joined the Safer Technologies 4 Schools (ST4S) Product Badge Program in 2024. The annual ST4S assessment is part of our commitment to supporting the online security and safety of students and schools. Learn more at st4s.edu.au

Outcomes

OUTCOME	KEY KNOWLEDGE	LOCATION
Unit 1 Area of Study 1 Outcome 1	On completion of this unit the student should be able to interpret teacher-provided solution requirements and designs, analyse data and develop data visualisations to present findings.	Chapter 1 and Chapter 2
KK1.1.1	• types and purposes of qualitative and quantitative data	pp. 3–5; 8–9
KK1.1.2	• characteristics of data types relevant to the selected software tools	pp. 15–16
KK1.1.3	• factors affecting the quality of data and information	pp. 12–15
KK1.1.4	• characteristics of data and information	pp. 17–18
KK1.1.5	• techniques for applying the Australian Privacy Principles (APPs) in the <i>Privacy Act 1988</i> (Cwlth) relating to the use, management and communication of data and information	pp. 18–25
KK1.1.6	• ethical issues arising from the management and communication of data and information	pp. 25–33
KK1.1.7	• referencing primary and secondary data and information using the American Psychological Association (APA) referencing system to acknowledge intellectual property	pp. 10–12
KK1.1.8	• characteristics of functional and non-functional requirements, constraints and scope	pp. 77–80
KK1.1.9	• design tools for representing the functionality and appearance of databases, spreadsheets and data visualisations	pp. 80–101
KK1.1.10	• structural characteristics of relational database management systems (RDBMS)	pp. 40–6
KK1.1.11	• structural characteristics of spreadsheets	pp. 40–6
KK1.1.12	• software functions and techniques for efficiently and effectively manipulating, validating and testing data to develop databases, spreadsheets and data visualisations	pp. 46–61
KK1.1.13	• use of spreadsheets to calculate descriptive statistics for analysis	pp. 46–8
KK1.1.14	• purposes of data visualisations for educating, entertaining, informing and persuading audiences	pp. 61–5
KK1.1.15	• types of data visualisations	pp. 65–6
KK1.1.16	• components of data visualisations	pp. 66–73
KK1.1.17	• formats and conventions suitable for databases, spreadsheets and data visualisations	pp. 73–7
Unit 1 Area of Study 2 Outcome 2	On completion of this unit the student should be able to interpret teacher-provided solution requirements to design and develop a software solution using an object-oriented programming language.	Chapter 3 and Chapter 4
KK1.2.1	• emerging trends in programming languages and artificial intelligence-based (AI) code generators for the development of software solutions	pp. 112–13
KK1.2.2	• characteristics of data types	pp. 119–22
KK1.2.3	• types of data structures	pp. 122–3
KK1.2.4	• characteristics of functional and non-functional requirements, constraints and scope	pp. 113–19
KK1.2.5	• design tools for representing the functionality and appearance of solution designs	pp. 123–6

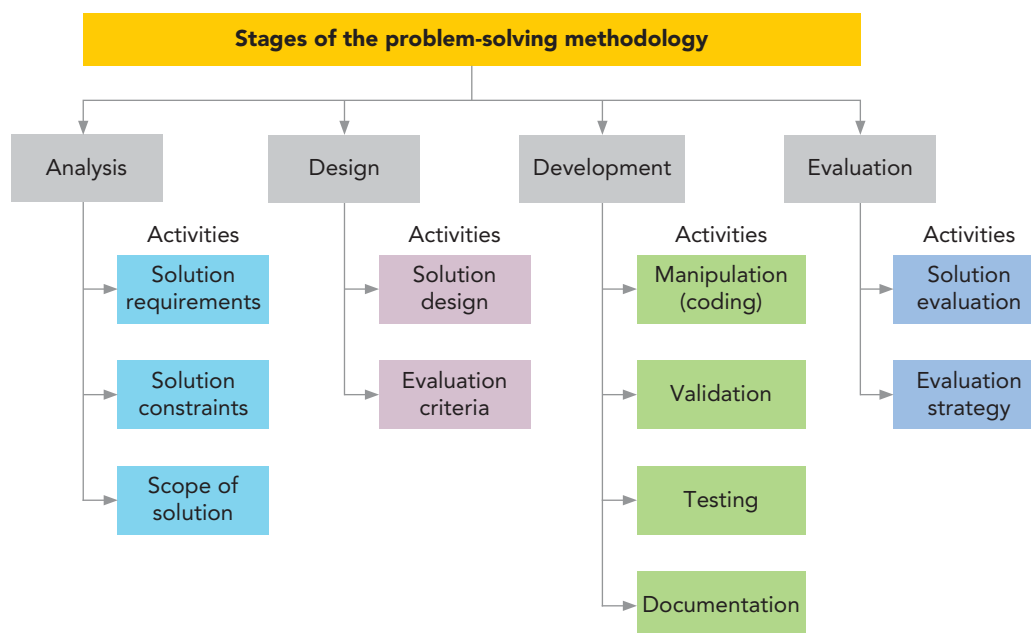
OUTCOME	KEY KNOWLEDGE	LOCATION
KK1.2.6	key legal requirements relating to intellectual property and copyright while designing and developing software	pp. 118–19
KK1.2.7	principles of OOP	pp. 145–8
KK1.2.8	features of a programming language	pp. 134–45
KK1.2.9	naming conventions for solution elements	pp. 148–9
KK1.2.10	purposes of internal documentation	pp. 149–51
KK1.2.11	validation techniques for data	pp. 151–2
KK1.2.12	debugging and testing techniques for checking software solutions function correctly	pp. 152–60
Unit 2 Area of Study 1 Outcome 1	On completion of this unit the student should be able to, in collaboration with other students, identify a problem, need or opportunity to analyse, design, develop and evaluate an innovative solution.	Chapter 5 and Chapter 6
KK2.1.1	the role of curiosity, ingenuity and the United Nations’ Sustainable Development Goals to inspire and drive innovation	pp. 175–7
KK2.1.2	functions and capabilities of digital systems used by individuals and organisations	pp. 178–89
KK2.1.3	the impact of emerging technologies	pp. 189–99
KK2.1.4	characteristics of creative and innovative solutions	pp. 222–3
KK2.1.5	solution specifications	pp. 223–6
KK2.1.6	methods for collecting data to determine user needs and requirements	pp. 221–2
KK2.1.7	design tools and techniques for representing solution designs	pp. 229–32
KK2.1.8	techniques for developing innovative solutions	pp. 226–8; 233–8
KK2.1.9	techniques for documenting the development of solutions, task delegation and monitoring project progress	pp. 217–21
KK2.1.10	techniques for testing and evaluating innovative solutions	pp. 238–46
KK2.1.11	how emerging technologies are affected by key legislation and frameworks	pp. 206–13
KK2.1.12	ethical issues arising from the development of emerging technologies	pp. 214–15
KK2.1.13	ethical issues arising from the use of artificial intelligence	pp. 215–16
KK2.1.14	evaluation criteria and techniques for evaluating the efficiency and effectiveness of innovative solutions	pp. 247–50

OUTCOME	KEY KNOWLEDGE	LOCATION
Unit 2 Area of Study 2 Outcome 2	On completion of this unit the student should be able to respond to a teacher-provided case study to analyse an organisation's software development practices, identify and evaluate current security controls and threats to software development practices, and make recommendations to improve practices.	Chapter 7
KK2.2.1	emerging trends in cyber security	pp. 297–9
KK2.2.2	functions and characteristics of key hardware and software components of networks required for communicating and storing data and information	pp. 260–8
KK2.2.3	strengths and limitations of wired, wireless and mobile communications technology	pp. 268–76
KK2.2.4	technical underpinnings of intranets, the internet and virtual private networks	pp. 276–80
KK2.2.5	applications and capabilities of Local Area Networks (LANs), Wide Area Networks (WANs) and Wireless Personal Area Networks (WPANs)	pp. 261–4
KK2.2.6	risks of using networks in a global environment	pp. 281–2
KK2.2.7	technical underpinnings of malware that can intentionally threaten the security of networks	pp. 289–90
KK2.2.8	security threats to data and information on networks	pp. 287–92
KK2.2.9	practices for reducing risks and mitigating threats to networks	pp. 282–94
KK2.2.10	cryptographic techniques to protect data and networks	pp. 283–7
KK2.2.11	emergence of artificial intelligence in providing network security mechanisms	pp. 280–1
KK2.2.12	the role of ethical hacking, such as identifying vulnerabilities and weaknesses in networks	pp. 294–5
KK2.2.13	key legislation and industry frameworks that affect how organisations ethically control the security and communication of data and information	pp. 295–6

Reproduced from the VCE Applied Computing Study Design (2025–2028), pp. 24–8, 31–5, © VCAA.

Problem-solving methodology

When an information problem exists, a structured problem-solving methodology is followed to ensure that the most appropriate solution is found and implemented. For the purpose of this course, the problem-solving methodology has four key stages: analysis, design, development and evaluation. Each of these stages can be further broken down into a common set of activities. Each unit may require you to examine a different set of problem-solving stages. It is critical for you to understand the problem-solving methodology because it underpins the entire VCE Applied Computing course.



Reproduced from the VCE Applied Computing Study Design (2025–2028) © VCAA; used with permission.

FIGURE 1 The four stages of the problem-solving methodology and their key activities

Analyse the problem

The purpose of analysis is to establish the root cause of the problem, the specific information needs of the organisation involved, limitations on the problem and exactly what a possible solution would be expected to do (the scope). The three key activities are:

- 1 identifying solution requirements – attributes and functionality that the solution needs to include, information it must produce and data needed to produce this information
- 2 establishing solution constraints – the limitations on solution development that need to be considered. Constraints are classified as economic, technical, social, legal and related to usability
- 3 defining the scope of the solution – what the solution will and will not be able to do.

Design the solution

During the design stage, two to three potential design ideas based on both appearance and function are planned and the most appropriate of these is chosen. Criteria are also created to select the most appropriate ideas and to evaluate the solution's success once it has been implemented. The two key design activities are:

- 1 creating the solution design – it must clearly show a developer what the solution should look like, the specific data required and how its data elements should be structured, validated and manipulated. Tools typically used to represent data elements could include data dictionaries, input–process–output (IPO) charts, object descriptions, flowcharts and pseudocode. The following tools are also used to show the relationship between various components of the solution: annotated diagrams, mock-ups, storyboards, site maps, queries, context diagrams, data flow diagrams and use case diagrams. Furthermore, the appearance of the solution, including elements such as a user interface, graphic representations or data visualisations, needs to be planned so that overall layout, fonts and their colours, for example, can be represented. Layout diagrams, annotated diagrams or mock-ups usually fulfil this requirement. A combination of tools from each of these categories will be selected to represent the overall solution design. Regardless of the visual or functional aspects of a solution design, at this stage a test for the design to ultimately ensure the solution is functioning correctly must also be created
- 2 specifying evaluation criteria – during the evaluation stage, the solution is assessed to establish how well it has met its intended objectives. The criteria for evaluation must be created during the design stage so that all personnel involved in the task are aware of the level of performance that ultimately will determine the success or otherwise of the solution. The criteria are based on the solution requirements identified in the analysis stage and are measured in terms of efficiency and effectiveness.

Develop the solution

The solution is created by the developers during this stage from the designs supplied to them. The 'coding' takes place, but also checking of input data (validation), testing that the solution works, and the creation of user documentation. The four activities involved with development are:

- 1 manipulating or coding the solution – the designs are used to build the electronic solution. The coding will occur here and internal documentation will be included where necessary
- 2 checking the accuracy of input data by way of validation – manual and electronic methods are used; for example, proofreading is a manual validation technique. Electronic validation involves using the solution itself to ensure that data is reasonable by checking for existence, data type and that it fits within the required range. Electronic validation, along with any other formulas, always needs to be tested to ensure that it works properly

- 3 ensuring that a solution works through testing – each formula and function, not to mention validation and even the layout of elements on the screen, needs to be tested. Standard testing procedures involve stating what tests will be conducted, identifying test data, stating the expected result, running the tests, stating the actual result and correcting any errors
- 4 documentation to support the development of the solution, including internal documentation. Internal documentation supports the functioning, maintaining and upgrading of the solution.

Evaluate the solution

At some time after a solution has been in use by the end user or client, it needs to be assessed or evaluated to ensure that it has been successful and does actually meet the user's requirements. The two activities involved in evaluating a solution are:

- 1 working out an evaluation strategy – creating a timeline for when various elements of the evaluation will occur and how and what data will be collected (because it must relate to the criteria created in the design stage)
- 2 evaluating the solution – providing feedback to the user about how well the solution meets their requirements, needs or opportunities in terms of efficiency and effectiveness. This is based on the findings of the data gathered at the beginning of the evaluation stage when compared with the evaluation criteria created during the design stage.

Unit 1

INTRODUCTION

VCE Unit 1 of Applied Computing looks at how software tools such as databases and spreadsheets can be used to create visualisations of data. You will also study programming languages.

Throughout the unit, you will apply the stages of the problem-solving methodology. You will analyse data and develop data visualisations as well as design and develop software solutions. Different types of data will be acquired and manipulated in database and spreadsheet software.

There are two outcomes to be completed in Unit 1.

Area of study 1 – Data analysis

OUTCOME 1 Your teacher will provide you with solution requirements and designs for which you need to gather and organise appropriate data, analyse it and present the findings as data visualisations. As part of the solution development you will need to validate your data and apply appropriate formats and conventions to the data visualisations.

Area of study 2 – Programming

OUTCOME 2 You will be provided with a set of program requirements by your teacher. You are to use these specifications to design and develop a solution created in a programming language. Validation, debugging and testing will be required as part of your development.



Data analysis

KEY KNOWLEDGE

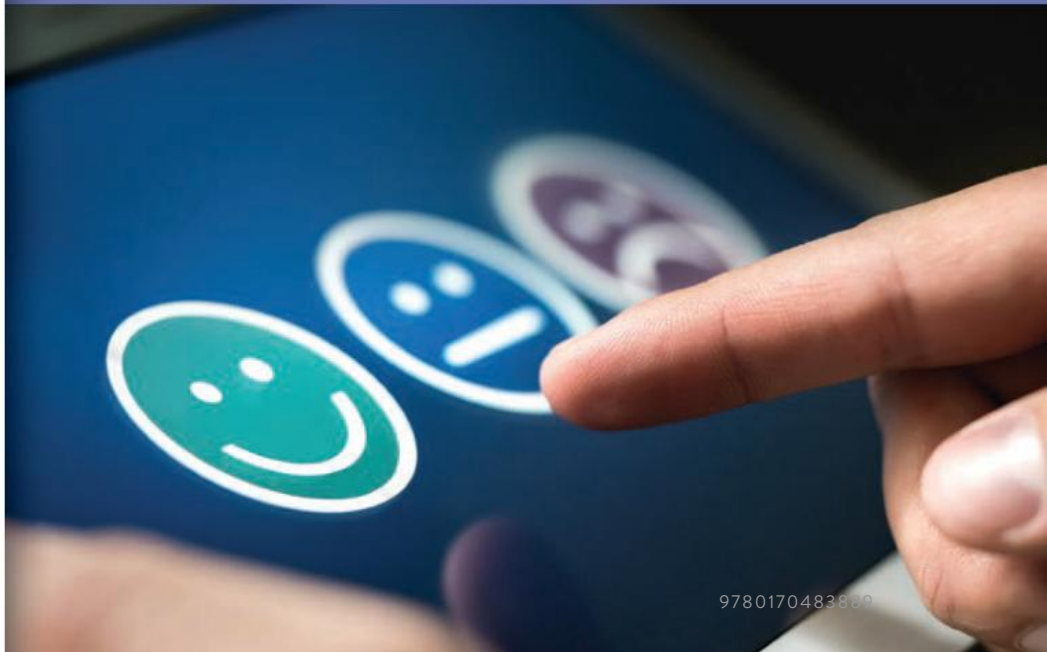
On completion of this chapter, you will be able to demonstrate knowledge of:

- KK1.1.1: types and purposes of qualitative and quantitative data
- KK1.1.2: characteristics of data types relevant to the selected software tools
- KK1.1.3: factors affecting the quality of data and information
- KK1.1.4: characteristics of data and information
- KK1.1.5: techniques for applying the Australian Privacy Principles (APPs) in the *Privacy Act 1988* relating to the use, management and communication of data and information
- KK1.1.6: ethical issues arising from the management and communication of data and information
- KK1.1.7: referencing primary and secondary data and information.

Victorian Curriculum and Assessment Authority, Victorian Certificate of Education Applied Computing Study Design 2025–2028, VCAA, 2024, p. 25

FOR THE STUDENT

In this chapter, you will respond to a teacher-provided analysis of requirements and designs to identify and collect data in order to present your findings as data visualisations. This chapter is based on Unit 1, Area of Study 1, and, together with Chapter 2, provides the key knowledge required to complete Unit 1, Outcome 1. At the end of Chapters 1 and 2, you should be able to interpret teacher-provided solution requirements and designs, collect and manipulate data, analyse patterns and relationships, and develop data visualisations to present findings.



Understanding research

Some people consume research, and others produce research. Consumers of research spend a lot of time reading other people's research rather than conducting their own. On the other hand, producers of research investigate or explore an area that has relevance to them, interpret their **data** and then communicate their findings.

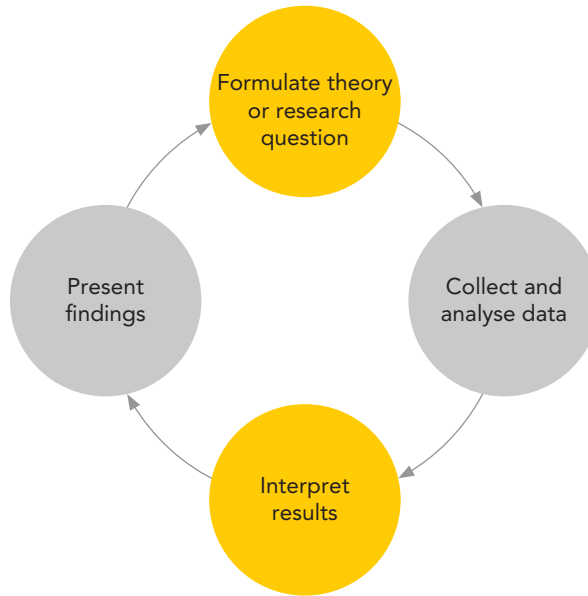


FIGURE 1.1 Producers of research often start with a theory or a research question.

Theories are usually general statements that describe something, provide an explanation of why something happens, and can be applied to predict what will happen in the future. Theories are, in principle, falsifiable or disprovable; that is, they contain information about the sorts of events that, if they were to happen, would show the theory to be false.

Some research questions are tied closely with theories. **Research questions** assist researchers to narrow the focus of the topic of the investigation. For example, 'Is therapeutic exercise of benefit in repairing cognition impairment in octogenarians?'

Hypotheses, on the other hand, are based on probabilities about what will happen according to the applied theory. Theories are tested by using data-collection tools such as surveys and/or interviews, and then the results of the study will either support or refute the hypothesis.

Types of research

Investigating or researching topics of interest may be undertaken using quantitative or qualitative research methods.

Quantitative data is measurable and specific, and is therefore relatively easy to chart or graph. At a simplistic level, quantitative data gathering is based on verifying a research question through the use of statistics and data that is largely numerical, while qualitative data provides a more in-depth understanding.

The following is an example of quantitative data:

56.0 per cent said a lack of role models was a significant or moderate obstacle to their career advancement.

Professionals Australia (2018). *All Talk: Gap between policy and practice a key obstacle to gender equity in STEM – 2018 Women in STEM Professions Survey Report*. August 2018, page 32.

SPSS and Minitab are statistical software packages used to analyse quantitative data.

When data has been gathered using surveys, focus groups, observation or other methods, quantitative data can be analysed by using software such as Excel, the Statistical Package for the Social Sciences (SPSS) and Minitab. This takes time and often involves hours of data entry, depending on the complexity of the data-gathering instrument. For data gathering, online surveys such as SurveyMonkey, Qualtrics, Google Forms and Microsoft Forms allow users to create surveys and manage the collection and analysis of quantitative data.

Qualitative data is about qualities or attributes, and is much harder to measure than quantitative data. You can gather qualitative data using instruments such as interviews, focus groups, video footage and observation. Generally, qualitative data needs to be recorded accurately and transcribed at a later stage. Online survey software permits qualitative data to be entered through text boxes.

The analysis of qualitative data is quite different from that of quantitative data. With quantitative data, the researcher looks for themes or patterns through the use of numbers, while with qualitative data, the researcher establishes rich descriptions and finds themes through reading the text and classifying these themes. The following example of qualitative data gathering is more descriptive:

Survey respondents noted that a lack of role models and the lack of women both in the workplace and in senior roles were issues that impacted them. Respondents also noted that women in senior roles were not necessarily always positive role models and that in male-dominated workplaces and professions, career support and advancement for women could often depend on positive male role models.

Professionals Australia (2018). *All Talk: Gap between policy and practice a key obstacle to gender equity in STEM – 2018 Women in STEM Professions Survey Report*. August 2018, page 32.

NVivo is an example of a software package designed to help with the analysis of qualitative data. It lets a researcher sort and classify qualitative (non-numerical) information, identify relationships within it and examine whether it supports particular hypotheses.

Advantages and disadvantages of quantitative and qualitative data

Participants are more willing to be part of a quantitative study as it is less demanding of them. Often, quantitative studies use surveys, which can capture a large sample. Having a large sample size provides statistical validity, and helps to accurately reflect the characteristics, attitudes or views of the population. Data is interpreted, relationships are identified and findings are then communicated. Conversely, because surveys do not have a provision for probing the participants further, the answers provided do not have as much depth and are at times superficial. If too much information were provided, researchers would be overwhelmed by the amount of data collected and would not be in a position to analyse it. This can also be due to time and budgetary constraints.

Qualitative research provides for rich, in-depth studies of participants. Researchers can ask further questions, especially if something of interest arises. Generally, qualitative studies are small, and provide a narrative description of a sample group. Data-gathering tools can include interviews and focus groups. However, because the sample size is small and the sample is not very random, conclusions may not generalise readily to a larger sample size. Findings may be peculiar to a particular sample.

TABLE 1.1 Comparison of qualitative and quantitative data aspects

Aspect	Qualitative data	Quantitative data
Analysis techniques	Thematic analysis, content analysis	Statistical analysis, computational modelling
Insights	Provides in-depth understanding of user experiences and motivations	Provides broad and generalisable insights
Flexibility	Very flexible, easy to explore new ideas and areas	Limited flexibility, predefined variables must be determined
Advantages	Can gain rich, detailed data Helps to provide contextual understanding Very useful for exploring complex issues	Easy to analyse and visualise Allows for generalisations to be made Can handle large data sets easily and efficiently
Disadvantages	Very time-consuming to process and analyse the data collected Tends to result in subjective interpretations Can be difficult to make generalisations from findings	Can miss contextual elements and nuances Requires large sample sizes for accuracy Can be costly to collect the required amount of data
Used for	Understanding user behaviour and improving user experience	Measuring system performance, analysing usage patterns

Data and information

The terms ‘data’ and ‘information’ are often used interchangeably. Data refers to the raw, unorganised facts, figures and symbols. Data can also mean ideas or concepts before they have been refined. In addition to text and numbers, data also includes sounds and images (still and moving).

Information is produced when data is manipulated into a meaningful and useful form. This can be achieved by organising the data and presenting it in a way that suits the needs of the intended audience. The information produced can be used to inform, entertain or persuade an audience. When information makes it possible to identify individuals, protecting their privacy becomes a consideration.

Technically speaking, a datum is a single item of data; however, the term ‘data’ is commonly used and accepted as both the singular and plural forms of the word.

Primary and secondary data and information

There are many sources of data and information, and many methods and techniques used to collect it. These include:

- what people say in interviews, questionnaires/surveys, personal histories, biographies;
- images, audio recordings and other audio-visual materials;
- records generated for administrative purposes (e.g. billing, service provision) or as required by legislation (e.g. disease notification);
- digital information generated directly by the population through their use of mobile devices and the internet;
- physical specimens or artefacts;

- information generated by analysis of existing personal information (from clinical, organizational, social, observational or other sources);
- observations;
- results from experimental testing and investigations; and
- information derived from human biospecimens such as blood, bone, muscle and urine.

National Health and Medical Research Council.
National Statement on Ethical Conduct in Human Research 2007 (updated 2018), page 33.

Sources

Primary sources of data provide a first-hand account of a person, object, event or phenomenon. Many of the methods and techniques outlined in the *National Statement on Ethical Conduct in Human Research* are sources of primary data. Questioning people or surveying their opinions can provide different insights and more in-depth data than using information from **secondary sources**. The data will often be more up to date and can provide more unusual and important insights into issues, especially at the immediate local level, than secondary sources, which often present overall conclusions and general summaries.

Techniques and methods

Collecting data is usually done through methods such as surveys and interviews. While the results of surveys are easy to present graphically, interview results often can only be presented as written summaries and conclusions. However, both require analytical discussions to interpret their meaning.

Surveys contain a range of questions that relate directly to the research question being investigated. Participants may answer these questions by selecting a response from a list of alternatives, such as A/B/C/D, selecting multiple options from a given list, using a Likert scale (either 1 to 5, very low to very high or strongly agree to strongly disagree), or in short or extended text-based responses.

Interviews are usually conducted face to face, sometimes in groups, and, depending on the number of interviews being conducted, it can take a substantial amount of time to conduct and analyse the responses. A major feature of an interview is the opportunity for in-depth follow-up and clarification questions. These are not possible with surveys, which are often answered in private. Interviews are very useful for eliciting people's feelings, attitudes and opinions, which are too complex to easily record in a survey.

Data collection methods

Before we can produce information, we first must start with data. Methods such as surveys, interviews or observation provide a means of capturing data. Sensor data provides for a device to collect physical measurements. Computers may automate that process and allow very fast or much slower recording over a longer time. Familiar examples are rainfall data automatically recorded at 15-minute intervals every day or heart rate measurements where a heart beats between 40 and 190 times per minute. These data points are collated, aggregated and averaged over a longer period of time to establish trends and patterns. The data collected can be used for a variety of purposes, including describing, predicting and improving processes within an organisation or for research.

A group interview is also known as a focus group.

Surveys

Surveys are common methods used to collect data. They can provide data about what the respondents think is true, or their preferences for consumer goods and political parties. A survey can be a quick way of gathering large amounts of data. Surveys need to be carefully designed, otherwise the participants' responses may not provide suitable data to analyse, rendering them useless. Questions used in a survey must be carefully worded so that the response will provide meaningful and useful data without the need for further clarification.

Interviews

Interviews are used to elicit people's opinions and beliefs. They can be used to gather data for research projects. Interviews are usually conducted with one or more participants in a quiet and relaxed atmosphere. They should be recorded, with the interviewee's permission, with easily used and unobtrusive audio or video equipment. Writing down the responses during the interview is not helpful to the interviewer or the interviewee. These records are research data in themselves, but may also be transcribed later. Collating and analysing information can be difficult and time-consuming and may require the use of someone with expertise. There are many interview styles that can be used:

- structured interviews, which follow a set list of questions
- semi-structured interviews, which follow a list of issues to be explored
- unstructured interviews, which involve spontaneous generation of questions and where the interview is driven by the interviewee rather than the interviewer.

Closed and open-ended questions

Questions used on a survey and during an interview can be closed or open-ended.

Closed questions limit the responses available to the respondent (Figure 1.2, page 8). They include 'Yes/No' boxes, multiple-choice questions, and scales on which attitudes and beliefs are measured using responses such as 'strongly agree', 'agree', 'disagree' or 'strongly disagree'.

Closed (or closed-ended) questions are generally considered to be quantitative in nature. They are called 'closed' because the range of answers the participant can choose is limited. They are considered to be quantitative because the response options can be converted to numbers. For example:

How often do you wash your car?

- 5 I always wash my car.
- 4 I sometimes wash my car.
- 3 I occasionally wash my car.
- 2 I wash my car once in a while.
- 1 I never wash my car.

Each of these options can have a value placed next to it. However, we do not talk in numbers and we shouldn't create surveys that only have numbers. Surveys should be thought of as a conversation between the person asking the questions and the person answering them.

Open-ended questions do not limit the answers that the respondent can give (Figure 1.3). They should be worded so that the responses received are capable of correct interpretation. For instance, if you asked the question, 'How do you feel about

1.1 THINK ABOUT APPLIED COMPUTING

What types of charts could you use to display the aggregated responses to closed questions?

Closed questions	
1	How long have you shopped at this store?
2	How many times per week do you go shopping?
3	How much do you spend per week?
4	Which of the following sources of information most influences your purchasing habits?
	• Advertising pamphlets delivered to the home
	• Newspaper advertisements
	• Television promotions
	• Recommendations from friends
5	Do you use a computer? ☐ Yes ☐ No
6	What time of day do you normally go shopping?

FIGURE 1.2

Closed questions should be designed to elicit short, straightforward answers.

Open-ended questions	
1	What is your opinion of the games available from this store?
2	How influential do you think the advertising campaign has been?
3	What are some of the errors in data entry that you have observed?
4	Describe the most frustrating experience you have had when using the computer system.
5	What are some of the problems you experience in receiving information on time?
6	What changes would you recommend to improve the billing system?

FIGURE 1.3

Open-ended questions try not to limit the answers the respondent can give.

the widespread use of computer games?', the responses would probably be too broad to be usefully categorised and analysed. The wording must therefore limit the scope of the possible responses to specific areas of interest: 'How has the playing of computer games affected your child's school results?' Open-ended questions also allow for follow-up questions, which are called probing questions, such as 'Why?' or 'Please give an example'. Such questions tend to elicit more detail.

Open-ended questions are 'open-ended' because participants are free to answer in any manner they choose. Unlike closed questions, there are no response options specified. They are qualitative because responses are considered and measured by feel rather than by numbers.

Closed questions are easier to develop, quicker to administer and answer, easier to collate and analyse, and can provide a large and balanced sample; however, they may not be useful for complex issues. In this case, open-ended questions may be needed as they elicit greater detail in the responses, can bring forth unusual ideas and can show links between various aspects of the issues.

Purposes of quantitative and qualitative data

Once data is collected, it is in a raw state. Raw data often requires scrubbing or cleansing to fill in unintentional blanks, correct errors (spelling, incorrect entry fields, misreading of question, etc.) and reformat consistent with required data types. Once cleaned, the data can be presented as lists or tables and is called primary data in a form closely similar to the original data. There are verified sources of census data available, with minimal editing of the original data. Similar demographic data sources allow researchers to perform their own manipulations and analyses to show trends and patterns extracted from the data.

Census data

The Australian Bureau of Statistics (ABS) was established under the *Census and Statistics Act 1905*.

The *Census and Statistics Act 1905* (CSA) provided the Australian Statistician with the authority to conduct statistical collections, including the Census of Population and Housing, and, when necessary, to direct a person to provide statistical information.

Australian Bureau of Statistics, 'Legislative framework'. Source: <https://www.abs.gov.au/about/legislation-and-policy/legislative-framework>

The *Australian Bureau of Statistics Act 1975* allowed for administrative operations to be updated. The ABS conducts an Australia-wide census every five years. Unlike other sampling surveys, a census surveys the entire population of Australia, including visitors to the country.

Every five years, the ABS counts every person and household in Australia. We call this the Census of Population and Housing.

The Census is the most comprehensive snapshot of the country and tells the story of how we are changing. Census data tells us about the economic, social and cultural make-up of the country.

The Census collects information from people, and in this way, it's like a big survey. However, surveys only select a sample of people to participate, whereas the Census includes everybody in the population.

Australian Bureau of Statistics, 'The Australian Census'. Source: <https://www.abs.gov.au/census/about-census/australian-census>

Census data serves as a foundational resource for demographic and statistical analysis. This data encompasses detailed information on population characteristics such as age, gender, ethnicity, education, income and employment. Researchers and policymakers utilise this comprehensive data set to identify demographic trends, understand population distributions and analyse social and economic conditions across the country.

Demographic data from the census is an important data set for urban planning, public policy development and resource allocation. For example, urban planners use population density and distribution information to design infrastructure and services that meet the needs of communities. Similarly, policymakers rely on demographic trends to formulate and evaluate policies in areas such as health care, education and social welfare, ensuring targeted and effective delivery of services. Additionally, businesses leverage this data for market analysis, helping them understand consumer demographics and economic conditions to inform strategic decisions.

Data generated by artificial intelligence

AI systems generate extensive amounts of quantitative data through applications involving numerical analysis, pattern recognition and statistical modelling. Examples include sensor data from devices, which produce continuous streams of metrics like temperature readings and motion detection, and predictive analytics, where AI algorithms use historical data to forecast future trends such as sales and risk assessments. Additionally, digital platforms leverage AI to collect user interaction metrics, including click-through rates and session durations, providing valuable numerical insights into user behaviour.

AI also processes and generates qualitative data, particularly through large language models (LLMs), natural language processing (NLP) and computer vision applications. For example, AI can analyse textual data from social media and customer reviews to identify themes, sentiments and trends, offering insights into public opinion and user experiences. In visual content analysis, AI interprets images and videos to extract information about objects,



scenes and activities, such as detecting unusual behaviour in surveillance footage. Similarly, AI processes audio data to transcribe and analyse speech, capturing qualitative aspects like tone and emotion.

More comprehensive and nuanced analysis of both qualitative and quantitative data can be achieved when using AI, particularly when integrating the two types of data. For example, an AI system might analyse customer reviews to quantify sentiment scores and identify recurring themes, then link these findings to sales data to understand the impact of customer sentiment on business performance. This holistic approach enhances decision-making across various fields by providing deeper insights.

Referencing primary sources

Once primary data has been gathered, details need to be carefully recorded to enable appropriate **referencing**.

Interviews

For an interview, the following details need to be documented so that an interested person can go back to the source for checking, clarification and further information:

- name of interviewee
- date of interview
- place of interview
- qualification to be an interviewee – that is, whether the interviewee is a **stakeholder** in the issue and/or an expert about it
- organisation to which the interviewee belongs (if relevant)
- contact information for interviewee – phone number, address, email address, online chat handle
- how the interview was conducted; for example, in person, by phone, email or online chat
- name and contact details of interviewer.

Surveys

If you want to cite an individual response to a survey, you need to record these details:

- name of respondent
- when the survey was completed
- title of survey
- organisation to which the survey belongs (if relevant)
- how the survey was conducted – paper/online.

Observation

For observation, the following details need to be recorded:

- the name of the person or group of people observed
- when the observation was conducted (date/time)
- where the observation was conducted.

Examples of referencing

Citations in a document help readers to find the source of the information and also assist students to avoid **plagiarism**. There are many ways to cite sources, such as providing footnotes or in-text citations, or listing sources at the end of the document in a bibliography or reference list.

Footnotes

Footnotes are listed at the bottom of the page on which a citation is made. Some academic disciplines prefer to use footnotes (notes at the foot of the page) to reference their writing. Although this method differs in style from the ‘author, date’ system, its purpose – to acknowledge the source of ideas, data or quotations without undue interruption to the flow of the writing – is the same. Footnotes are usually sequenced: series of numbers above the text (superscript) are placed in the appropriate part of the text to indicate the cited work and are matched at the bottom of the page to the footnote. A footnote lists the author, title and details of publication, in that order. For example, in a 2018 report, *Optimising STEM Industry–School Partnerships: Inspiring Australia’s Next Generation Final Report*, the paragraph shown in Figure 1.4 appears in the introduction.

To meet the challenges of the future, as noted in the ISA 2030 Plan, governments, educators and industry must work more closely together to maximise and amplify the impact of industry investment and ensure students and teachers keep up with the rapid pace of change in STEM disciplines.¹

1 Innovation and Science Australia 2017, *Australia 2030: prosperity through innovation*, Australian Government, Canberra, p. 33.

FIGURE 1.4

An example of a footnote. A superscript number is inserted in the body of the text, and the full reference is provided at the bottom of the same page.

APA

The American Psychological Association (APA) created a style guide to assist with academic writing such as essays, books and other publications. The APA style is widely used and is the reference style that students are expected to use in VCE Applied Computing. Citations within the text, and their corresponding source details in a reference list at the end of the work, are necessary elements of the APA style. These show the reader where ideas and research have come from. Typically, when referencing using APA, the author’s surname and the date of the publication are featured in the text. If quoting directly from the source, then the page number is also included. Figure 1.5 shows an example.

The Theory of Individual Differences (Trauth et al., 2004) draws on previous research on gender and IT and emphasises the differences within genders rather than between them. The theory ‘looks at men and women as individuals, who experience a range of different social-cultural influences which shape their inclination to participate in IT in a variety of ways’ (Trauth et al., 2004, p. 116).

...

Trauth, E. M., Quesenberry, J. L., & Morgan, A. J. (2004). *Understanding the under representation of women in IT: Toward a theory of individual differences*. Paper presented at the SIGMIS 2004 Conference on Computer Personnel Research, Tucson, Arizona, USA.

FIGURE 1.5

An example of APA-style citation. The author’s name, date of publication and sometimes the page number are inserted in the body of the text; the full reference is provided in the reference list at the end of the document.

Within the main body of the document, any reference to the publication will need to be cited. As illustrated in Figure 1.5, when using the APA style, the format usually follows

author–date–page, where the author’s last name, the year of publication and the page number of the quote are referenced. These are all separated by commas, and are placed within parentheses following the text. The page number is preceded by a lower case ‘p’ with a period (full stop) after it; for example, ‘(Trauth et al., 2004, p. 116)’.

Interpretation of information for communication and decision-making

Data and information can be collected, stored and interpreted for a particular purpose or activity, including research. The findings or results may be of significant benefit to the participants or their relatives, as in the case of the analysis of human biological specimens; conversely, it is worth noting that there could be psychological, social or legal risks involved.

The purpose of disseminating outcomes from the interpretation of the information is to make a contribution to knowledge. Sometimes, the interpretation of findings assists with decision-making to better inform participants about options available. Common ways to disseminate findings are through publications such as journals, books, conference presentations, data visualisations, creative works and performances.

‘The approach taken to communicating findings and results should ... adhere to the ethical principles of justice, respect and beneficence ... including consideration of the values and preferences of traditional custodians, such as Aboriginal and Torres Strait Islander peoples.’

National Health and Medical Research Council. *National Statement on Ethical Conduct in Human Research 2007 (updated 2018)*, page 38.

Quality of data and information

One aspect of identifying relevant data from a given data set is ensuring that the data is usable. To be usable, data must be relevant, accurate, free from bias and reliable.

Accuracy

Data that is collected must be **accurate**, otherwise the insights will be poor and decisions based on this data could cause integrity issues. Transcription is often a cause of error. Transcription errors occur when the person entering the data misreads the information through, for example, a lapse in concentration or being interrupted, or presses the wrong key. It is easy to make a mistake when entering a large amount of data, particularly numbers with many digits that may not contain spaces or punctuation to signify thousands. Clearly, if the data collected is incorrect, the information produced from it will be incorrect. If data has been gathered from a primary source, it is a good idea to check the data against the source. If data has been gathered from a secondary source and is suspect, it is worthwhile verifying the data using other secondary sources.

Freedom from bias

Bias can easily creep into data and make the information processed from it unreliable. Several influences can lead to the introduction of bias into data; these include vested interests, timing and small sample sizes. There can also be bias through sorting and bias through graphic representations.

Vested interest

Bias can enter data if the respondent to a survey or interview has a **vested interest** in the outcome of the research. A common example is seen in the use of celebrities who are paid to promote particular products in commercials or social media. It would be unreasonable to

trust their assertions that one product is better than others simply because they are celebrities; they are only saying what they have been paid to say and may not necessarily be providing an independent judgement that has been derived from research or experience.

Timing

The timing of the data collection may also introduce bias. Suppose, for example, you plan to survey a sample of the population for their views about Australia becoming a republic. The data you gather may be biased if, just before the survey is conducted, a royal tour takes place and there is extensive media coverage about the royal family. The timing of the data collection would introduce bias because it coincides with a significant event that could influence people's responses.

Note too that bias is not restricted to data gathered from surveys or during interviews. For example, suppose that Qantas needed to decide whether to schedule two new weekly flights to New York. The decision could depend on the demand for existing flights. If the airline collected data from bookings made over a four-week period just before or during a significant event, such as the New York marathon, the data gathered would be biased. Such data should not be relied on for making this decision because the influence of this event on customer demand is irregular and unlikely to occur again.

Small sample size

Choosing a sample size that is too small may also create bias. The sample size and composition must be suitable for the purpose of the data collection. Usually, a larger sample size leads to greater **precision**, provided the sample composition is suitably representative of the target population. The sample size must be big enough to make any conclusions drawn, and information produced, credible. For example, if you wanted to determine whether or not the school uniform should be changed, the data would not be reliable if you only surveyed students in one class. Not only is this sample too small to be representative of the student body; it would also fail to include other stakeholders, such as parents and school administrators. Similarly, if you wanted to gather sales data over a four-day period to predict monthly sales at a fish and chip shop, this time scale would not be sufficient to make a prediction. For instance, by collecting data on the four Mondays in the month, you may be selecting the quietest trading days in the week. If you picked the four Fridays in the month, you may be picking the busiest trading days. When selecting a sample size, you need to ensure that it is representative of the whole population.

Bias through sorting

The way in which you sort lists can introduce bias, although this is frequently unavoidable. A classroom teacher often consults a class list that is sorted alphabetically – for example, to select students for special tasks. The list is biased towards students whose surnames appear early in the alphabet and are thus at the top of the class list. If you need to hire an electrician and consult an online directory, it is more likely that you will pick an early entry than one from the second page of listings. Bias of this type is difficult to avoid, so it is preferable to educate the user to recognise that the output has built-in bias and to encourage strategies to overcome that bias.

Bias in graphic representations

Bias can occur through your choice of typeface or the scale or size chosen for graphics. Graphic representations should be sized proportionally to avoid overstating or trivialising

the importance of one of the **variables** involved. For example, in Figure 1.6, a teacher has created a graph to show the heights of her students. The graph does not really give a clear picture of how the heights vary. The bars are all of similar height, so it is difficult to see the differences between them. In contrast, the different heights in Figure 1.7 are more distinct. This has been achieved by decreasing the scale of the vertical axis. The variation in the student heights looks much greater, even though the data has not changed. This graph makes it easier to see the small differences.

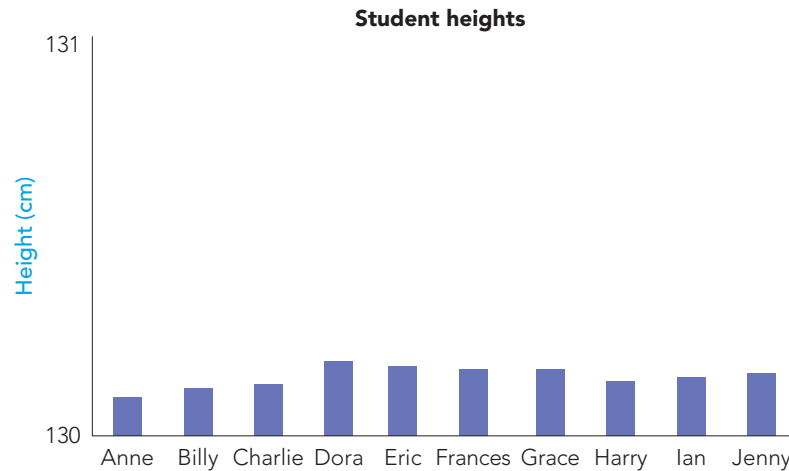


FIGURE 1.6 Column graph with little variation

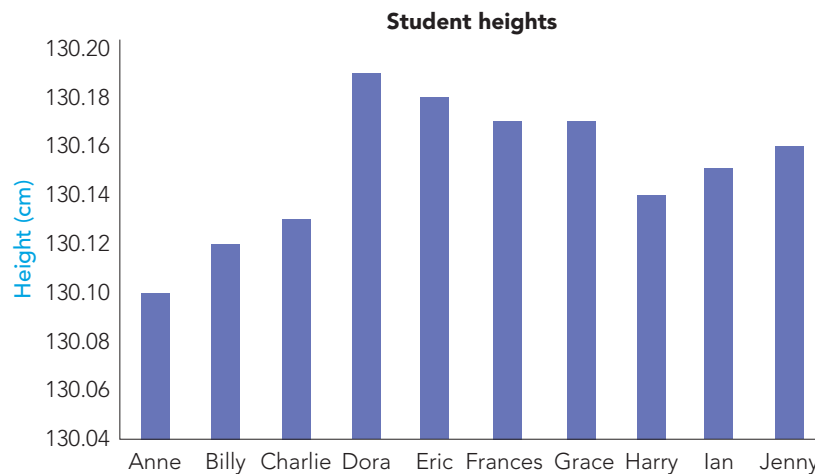


FIGURE 1.7 Column graph with greater variation

1.2 THINK ABOUT APPLIED COMPUTING

Create a list of five organisations in Australia that provide data sets for public use. They may be government departments or private organisations, or a mix of both.

Integrity

Data integrity refers to the quality of the data. An information system's purpose is to transform data into useful information. It is therefore important that the data input has integrity, because the information produced is based on the data itself. For example, if a customer database contains the wrong postal details for a customer, this is incorrect data. When a data set contains these types of errors, it loses its integrity. The more errors the data set contains, the lower its integrity. There are many ways in which the quality or integrity of data can be measured.

Relevance

To produce usable information, data must be **relevant**. For example, if a computing department in an organisation is evaluating PC-only software, then surveying people who only use an Apple device is not useful. The data collected from Apple users would not be relevant to the overall data collection. Data also needs to be processed while it is current, because decision-making should be based on current data, not outdated data.

Reliability

The internet has made it easier for people to communicate their views and present information in a format that is easily accessible to others. There are many personal websites, homemade videos, wikis, podcasts, vodcasts and a plethora of unchecked information on the internet, and some of the views presented may not be widely accepted or proven to be accurate. Alternatively, there are many sources on the internet that *are* **reliable**, such as the World Health Organization (WHO), the Smithsonian Institution, the Australian Bureau of Statistics, UN Data, Google Public Data Explorer, Eurostat and universities.

Data types relevant to selected software tools

Many different software tools can be used to create data visualisations; most, however, are derivatives of either a spreadsheet tool or a database tool. Data needs to be separated into distinct fields or columns that only have one data type in them. **Data types** are particular forms that an item of data can take. Common data types are text, numeric and Boolean. Each of these is discussed in the section that follows Table 1.2.

TABLE 1.2 Most common data types used in databases

Data type	Characteristics	Examples
Text	Alphanumeric	Name, address, postcode and phone number
Numeric	Numbers only (see below for different forms)	Any number to be used in a calculation
Boolean	Represents one of two states, True/False; can be used in logical tests	Also represented as Yes/No, On/Off and 1/0

Text: character, string

Character

A **character data type** holds a single letter, number or symbol. Numbers stored as a character data type cannot be used in mathematical calculations. This is a text field that will only accept a single alphanumeric character. It is used when there are multiple options for a value, but they can be represented with a single character to make data entry easier and to save storage space. For example, small, medium and large sizes of the same type of wooden box might be entered as 'S', 'M' or 'L', respectively.

THINK ABOUT APPLIED COMPUTING

1.3

Find three reliable sources and describe in one paragraph the type of data they produce.

Area of Study 1 concentrates on students presenting their work in ways that include database, spreadsheet and data visualisation solutions. This section will concentrate on data types relevant to databases and spreadsheets.

String

A text data type consists of a '**string**' of characters'. Text may also be referred to as a string, and the terms are sometimes used interchangeably to refer to the same data type. This type of field holds a mix of characters (letters, numbers, special characters), also referred to as alphanumeric. Names and addresses are considered to be text data. Also, postcodes and telephone numbers are normally formatted as text – even though they contain numerals – because they may contain spaces or begin with leading 0s and are not intended to be used in mathematical calculations. It is also more efficient to store the values as text rather than as a large numeric value.

Numeric: integer, floating point, date/time

The numeric data type refers to numbers that are to be used for a calculation of some kind. For example, the quantity of an item purchased might need to be multiplied by its price in order to calculate a total amount payable. Numeric fields can be separated into two distinct categories: integer and floating point.

Integer

An **integer** is a number without a fractional or decimal component – in other words, a whole number. Integers include negative numbers.

Floating point

A **floating-point number** is a number with a fractional or decimal component, such as a decimal number. Where decimal numbers are required, such as when dealing with financial transactions or percentages, then the floating-point data type is used. Examples of the floating-point data type are '3.17', '-6.890' and '0.001'.

A floating-point number can be split into two parts: the mantissa and the exponent. The **mantissa** holds all of the digits in the number, with a decimal point after the first digit that is not zero. The **exponent** holds the power of 10 by which the mantissa must be multiplied to regain the original number. For example, 0.00317 can be expressed as 3.17×10^{-3} , in which case the mantissa is 3.17 and the exponent is -3.

Date/time

Strictly speaking, a date format is another variation on the numeric data type. The value of a date is normally based on the number of days since the 'zero' day built into the operating system. For example, the day '1' might be displayed as 01 January 1900, while '42673' would be displayed as 30 October 2016. Calculations can be performed on dates, which can be handy when considering the difference between them. Dates can be formatted to show a combination of years, months, days, hours, minutes and seconds, depending on the needs of the user. In terms of time, they can also display 12- and 24-hour clocks.

Boolean

A **Boolean data type** can hold one of only two possible values, usually 'true' or 'false'. A Boolean data type is known as a logical data type. It is often used in conditional statements to test if a condition or value is true or not, but it may also represent Yes/No or On/Off.

THINK ABOUT APPLIED COMPUTING

1.4

In theory, a Boolean data type only requires 1 bit of memory space. Conduct research to find out why it actually needs to take up 1 byte of memory.

Characteristics of data and information

The characteristics of data and information refer to the qualities and attributes that determine their use, efficiency and effectiveness in various contexts. These characteristics influence how data and information can be collected, processed, analysed, interpreted and used for decision-making, problem solving and communication.

Size

Size refers to the amount or volume of data. This can range from a few data points (small data sets) to vast quantities of information (large data sets, often referred to as big data). Size affects storage requirements, processing capabilities and the complexity of analysis. For example, a few hundred rows in a spreadsheet are manageable with basic tools, whereas terabytes of data require more sophisticated systems for storage and analysis.

Structure

Structure relates to the organisation and format of data. Structured data is highly organised, typically in tables or databases, making it easily searchable and analysable. Unstructured data lacks a specific format, including things like text documents, videos and social media content. Semi-structured data includes elements of both, such as XML and JSON files. The structure influences how data can be accessed, manipulated and analysed.

Relevance

Relevance measures how pertinent or useful the data is to the specific needs or questions being addressed. Data is considered relevant if it directly supports the objectives of the analysis or decision-making process. Irrelevant data can distract or mislead, whereas relevant data provides valuable insights and supports informed decisions.

For example, sales data from the past year is relevant for forecasting next year's sales. However, employee satisfaction survey results might be less relevant for this specific task but highly relevant for improving workplace conditions.

Accessibility

Accessibility refers to how easily data can be obtained and used. Accessible data is available to the right people at the right time, with appropriate permissions and in a usable format. Factors affecting accessibility include data storage solutions, user access controls and the availability of data retrieval tools. For example, a cloud-based database accessible through an online portal is highly accessible, allowing authorised users to retrieve data from anywhere. In contrast, a physical ledger stored in a locked office is much less accessible, requiring physical presence and proper authorisation to access. High accessibility ensures that data can be efficiently leveraged for various purposes.

Clarity

Clarity describes how understandable and interpretable data is. Clear data is well defined, consistent and free of ambiguity, making it easier to analyse and use effectively. This includes having well-labelled data fields, consistent units of measurement and straightforward data formats. For example, a report with clearly labelled graphs and consistent terminology is clear and easy to understand. In comparison, a data set with ambiguous column names and mixed measurement units (e.g. mixing metres and centimetres) lacks clarity, making it difficult to analyse accurately. Clarity reduces errors and misunderstandings in data interpretation.

XML files are a format for structuring data using tags to define elements and attributes, making it both human-readable and machine-readable. JSON files use a lightweight, text-based format with key-value pairs to represent data structures, widely used for data interchange between web applications and servers.

Context

Context involves the background information and circumstances surrounding the data, providing essential details about its source, conditions of collection and relevant metadata. Context helps in understanding the significance and limitations of the data, ensuring accurate analysis and interpretation. For example, sales data from a retail store during a holiday season has a specific context that influences its interpretation, such as increased customer traffic and seasonal promotions. Without context, data can be misinterpreted or misapplied – in this case, by interpreting a spike in sales as a general trend rather than a seasonal event.

Privacy

When undertaking research, it is very important to uphold the privacy of the participants. Privacy is a fine balance between their interests and those of researchers. Privacy laws attempt to stop inappropriate intrusion into the lives of individuals. Often, however, the problem is not the collection of data, but how the data is used or misused by people entrusted with it. To maintain privacy, and to **de-identify** the data, personal identifiers such as names and birthdates that are associated with individuals need to be removed so that information cannot be traced or identified.

Care must be taken when de-identifying data. The example of de-identified data in Table 1.4 only removes the names and dates of birth of survey participants. Depending on the remaining data and the information generated, this may be insufficient to protect individual research participants from being identified (even accidentally) by at least some users of the information. This is particularly likely if the users know who the members of the sample group are, or if they are familiar with the wider population that is being sampled. Users of the information may be able to identify a survey participant even after his or her name and date of birth have been removed. This could be caused by the presence of a characteristic known to be rare in the sample group or wider population, because this helps to narrow possibilities. Combinations of rare characteristics would amplify this effect, so that users of the information could make an intelligent guess as to a person's identity.

TABLE 1.3 Data that has been collected but not de-identified

Given name	Surname	Height (cm)	Weight (kg)	Sex	Birthday	Postcode	Age	Active
Charlotte	Micelli	145	48	F	22/10/2014	3103	11	No
Candace	Fock	153	40	F	10/10/2014	3105	11	Yes
Abdul	Mohammad	160	37	M	19/10/2013	3122	12	Yes
Eric	Dorian	142	38	M	12/12/2014	3040	11	Yes
Jaspreet	Singh	148	41	M	31/12/2013	3041	12	No
Sienna	Megane	152	42	F	2/12/2014	3103	11	Yes
Elie	Wu	144	43	M	6/01/2014	3122	11	No
Jo	Wu	149	47	F	6/01/2014	3122	11	Yes
Hamish	DiPaola	154	37	M	2/02/2014	3044	11	No
Jacinda	Black	143	46	F	4/04/2014	3111	11	No

TABLE 1.4 Data that has been de-identified. Personal information such as birthdates, first names and surnames has been removed so that data cannot be identified or traced.

Participant	Height (cm)	Weight (kg)	Sex	Postcode	Age	Active
ID1001	145	48	F	3103	11	No
ID1002	153	40	F	3105	11	Yes
ID1003	160	37	M	3122	12	Yes
ID1004	142	38	M	3040	11	Yes
ID1005	148	41	M	3041	12	No
ID1006	152	42	F	3103	11	Yes
ID1007	144	43	M	3122	11	No
ID1008	149	47	F	3122	11	Yes
ID1009	154	37	M	3044	11	No
ID1010	143	46	F	3111	11	No

Australian Privacy Principles

Originally, the *Privacy Act 1988* (Cwlth) only dealt with the handling of data by Australian government agencies. Many people criticised this limitation because it seemed that private organisations were not required to apply even the most basic of safeguards on data they collected. Even worse, there were no regulations preventing non-government organisations from collecting data by any method and using it for any purpose without the consent of the people whose private details were concerned. In particular, the rapid growth of electronic transactions, especially over the internet, led many people to demand some sort of legal protection from those who might gather data about internet browsing habits. The government was keen to encourage the development of electronic commerce while protecting the confidentiality of consumers and increasing public confidence in electronic transactions. These amendments have now been incorporated into the *Privacy Act 1988* and are the most significant changes to privacy laws since the inception of the legislation.

There have been several additional powers included within this Act since 1988, but its main purpose has remained unchanged. The *Privacy Act 1988* was amended by the *Privacy Amendment (Enhancing Privacy Protection) Act 2012*. This came into effect in 2014. As part of this Act, the Australian Privacy Principles replaced the National Privacy Principles and the Information Privacy Principles so that Australia now has one set of privacy principles. The Australian Privacy Principles generally apply to Australian government agencies. They do not apply to local councils or state or territory governments. Some states have their own privacy laws. For example, Victoria has the *Privacy and Data Protection Act 2014* (PDPA).

The changes to the Privacy Act include 13 privacy policy principles known as the Australian Privacy Principles (APPs – see Table 1.5). The APPs were devised to set out the standards,

Mandatory data breach notification amendments have also been passed.

TABLE 1.5 Australian Privacy Principles**APP 1 Open and transparent management of personal information**

Ensures that APP entities manage personal information in an open and transparent way. This includes having a clearly expressed and up-to-date APP privacy policy.

APP 2 Anonymity and pseudonymity

Requires APP entities to give individuals the option of not identifying themselves, or of using a pseudonym. Limited exceptions apply.

APP 3 Collection of solicited personal information

Outlines when an APP entity can collect personal information that is solicited. It applies higher standards to the collection of 'sensitive' information.

APP 4 Dealing with unsolicited personal information

Outlines how APP entities must deal with unsolicited personal information.

APP 5 Notification of the collection of personal information

Outlines when and in what circumstances an APP entity that collects personal information must notify an individual of certain matters.

APP 6 Use or disclosure of personal information

Outlines the circumstances in which an APP entity may use or disclose personal information that it holds.

APP 7 Direct marketing

An organisation may only use or disclose personal information for direct marketing purposes if certain conditions are met.

APP 8 Cross-border disclosure of personal information

Outlines the steps an APP entity must take to protect personal information before it is disclosed overseas.

APP 9 Adoption, use or disclosure of government related identifiers

Outlines the limited circumstances when an organisation may adopt a government related identifier of an individual as its own identifier, or use or disclose a government related identifier of an individual.

APP 10 Quality of personal information

An APP entity must take reasonable steps to ensure the personal information it collects is accurate, up to date and complete. An entity must also take reasonable steps to ensure the personal information it uses or discloses is accurate, up to date, complete and relevant, having regard to the purpose of the use or disclosure.

APP 11 Security of personal information

An APP entity must take reasonable steps to protect personal information it holds from misuse, interference and loss, and from unauthorised access, modification or disclosure. An entity has obligations to destroy or de-identify personal information in certain circumstances.

APP 12 Access to personal information

Outlines an APP entity's obligations when an individual requests to be given access to personal information held about them by the entity. This includes a requirement to provide access unless a specific exception applies.

APP 13 Correction of personal information

Outlines an APP entity's obligations in relation to correcting the personal information it holds about individuals.

rights and obligations for collecting, handling, holding, accessing, using, disclosing and correcting personal information. The APPs oversee the handling of personal information by:

- Australian and Norfolk Island government agencies
- all private health service providers
- businesses that have an annual turnover of \$3 million or those that trade personal information.

It should be noted that Section 6 of the Privacy Act defines ‘personal information’ as ‘information or an opinion about an identified individual, or an individual who is reasonably identifiable’. This might include a person’s name and address, medical records, bank account details, photos, videos and even information about what an individual likes, their opinions and where they work. Another important aspect of the Privacy Act refers to ‘sensitive information’. Sensitive information is defined in the glossary and includes health information. The Privacy Act provides a higher level of privacy protection to sensitive information than to other personal information.

Of particular interest for this study are Principle 2, ‘Anonymity and pseudonymity’, Principle 6, ‘Use or disclosure of personal information’, and Principle 11, ‘Security of personal information’.

APP 2, ‘Anonymity and pseudonymity’

Australian Privacy Principle 2 provides individuals dealing with organisations the option of using a different name or a **pseudonym** in relation to a particular matter. This measure is in place so that individuals cannot be identified. In addition, individuals can also remain anonymous. For example, when an individual calls an organisation, often a message states that the call will be recorded for training purposes. If the individual objects, the call is not recorded. At a later date, when staff from the organisation receive training, the names of the individuals whose voices have been recorded must be changed in order to protect their identity when these real examples are used.

Anonymity of data may be of particular concern when deciding what must be removed to de-identify data sufficiently to protect all its potential users. It is also important when reporting personal information anonymously or using pseudonyms in a newspaper report, for example.

APP 6, ‘Use or disclosure of personal information’

Australian Privacy Principle 6 states that the information that is being held is in line with the primary purpose it was intended for. Information cannot be used for a secondary purpose unless the holders of the information have received consent from the individuals concerned. For example, if a sporting organisation collected information about its players for the purpose of organising competitions and making the details available to the coaches and captains, then that would be the primary purpose of the data. However, if the sporting organisation wanted to provide the details of its members to a shop that specialised in sporting merchandise, then it could not do so, as this would be using the information for a purpose for which it was not intended.

APP 11, ‘Security of personal information’

Australian Privacy Principle 11 refers to measures taken to actively secure personal information held, and also considers whether those who hold this personal information are permitted to retain it. Reasonable steps need to be taken to protect the information from misuse, interference and loss, as well as unauthorised access, modification or disclosure. Additionally, once there is no longer a need for any purpose, reasonable steps need to be taken to destroy or de-identify the personal information held.

Security to protect personal information

Data must be stored in a way that preserves confidentiality and meets all legal requirements. Stored data can be protected with both physical and software-based controls, such as backing up of data and shredding of confidential documents.

Data needs to be securely retained and then securely disposed of or destroyed when no longer needed. How long the data or material needs to be retained depends on the type of data. The minimum time frame for data to be retained for research purposes is five years from the date of any published or reportable outcome based on the data. For some specific types of research, other time frames may be required. For data such as identifiable health research data, the minimum period may need to be seven years. For identifiable health research data involving children, the data may need to be kept until the individuals involved turn 25, or five years after any research outcome based on the data is published or made available for release, whichever is the longer.

Physical security controls

Steps need to be taken to ensure that personal information is not inappropriately accessed. Hard copies of personal information need to be secured, and workspaces need to be designed to facilitate privacy. Consideration should be given to what measures are taken to control access. For example, swipe cards allow particular personnel into certain workspaces. Logs are kept of staff movements and their access to particular rooms, and security and alarm systems are used as a mechanism to control entry to the workplace. Those areas of an organisation where there is a greater need for access to personal information, such as human resource departments, should be physically segregated from other areas.

If the organisation requires an attendant to greet clients, consideration needs to be given to the placement of computers and computer monitors so that clients cannot see what is on the screen. Other measures include having designated areas for visitors so that they cannot see forms or hear telephone calls where personal information could be shared. Other physical controls include the provision of secure storage spaces near workstations to secure documents temporarily, and a place for securing physical files containing personal information. If files are placed in lockable cabinets, there needs to be consideration given to who has access and how access to keys is controlled.

Encryption

Encryption is the process of translating data into a code that can only be read by authorised users. To read an encrypted file, you must have access to a secret key that you use to **decrypt** the data. **Unencrypted data** is also known as 'plain text'. Data that has been collected for research purposes needs to be secured so that only authorised people have access. Encryption is one way to ensure that the data is secured. Encryption methods need to be reviewed periodically to ensure that they continue to be relevant and effective. Areas where encryption could be used include:

- databases that contain personal information
- servers, backups and cloud storage services
- mobile devices such as smartphones and tablets

- data storage devices such as flash drives and external hard drives
- data transferred on internal networks or over the internet – for example, using email or shared files.

Backups

Backups form an essential step in data management. Regular backups protect against a number of risks, including human error, computer crashes and software faults. Critical data files or data that are used regularly should be backed up frequently.

It is not unusual for someone to accidentally delete an important file or edit a document and later realise that some important information was removed. One strategy to minimise such loss of data by an authorised user is to maintain a backup system. Important files inadvertently lost can be retrieved from the backup media.

A **full backup** copies all of the files from a device to a storage medium. It can take considerable time and is usually performed once over a time period (such as a week, fortnight or month). A **differential backup** copies only those files that have been changed since the last full backup. Restoration of data would involve restoring files from the full backup and then from the differential backup. An **incremental backup** is similar to a differential backup, the difference being that it uses more than two backup media, while a differential backup uses only two media. An incremental backup only copies files that have been changed since the last incremental backup. It is the most complicated strategy from which to restore files since it requires restoration from a full backup and then from a series of incremental backups. It is a good practice to clearly label all backup media so that you know when the backup was made and what is on it.

Location of backup files

Once you have created backups, where do you put them? Ideally, your backups should be stored in a location that is safe from theft and damage caused by extremes of temperature or disasters. Most small businesses have a fireproof and waterproof safe in which valuable company documents are stored. This might also be used to store backups. It is preferable, however, to store backups at a remote location, perhaps even in the cloud. This means that if there is a large natural disaster, such as a flood or an earthquake, the backups will be safe.

One last point to remember is to ensure that backups actually work when you want to restore the data. It is important to test the effectiveness of your backup files by running a disaster recovery simulation. If files cannot be restored from the backup or the system refuses to recognise them, it is better to discover this before a real emergency.

Cloud-computing companies provide off-site storage, processing and computer resources to individuals and organisations. These companies are typically third party (i.e. not one of the individuals or organisations directly concerned with the data) and they store data to a remote database in real time. The internet provides the connection between this database and the user's computer. One of the advantages of cloud storage is the ability to access data from any location that has internet access, which eliminates the need to carry a USB flash drive or a hard drive to retrieve and store data. The ability to share files with other people and to collaborate simultaneously, such as by using Google Docs, is also an advantage. Finally, if something were to happen to the computer, such as a fire or natural disaster, and the data on it were to be destroyed, having the data saved off-site in the cloud would prevent the data from being lost.

Google Docs, Sheets, Slides and Forms allow users to create, edit and collaborate with documents, spreadsheets, presentations and surveys (Figure 1.8). Users can edit files using a Google application and work on them at the same time as others, so they can read or make edits simultaneously.

FIGURE 1.8

A selection of applications from Google's suite of office tools



Source: Google LLC

Username and passwords

A username is usually identifiable as belonging to a particular person and can be easily remembered. **Username**s are uniquely assigned to users. **Password**s are set by the user and should be known only to that user. To maintain high levels of security, user passwords should:

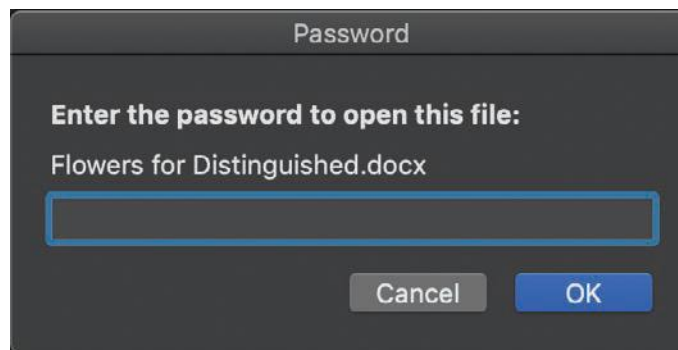
- be at least eight digits long
- include non-alphabetical characters
- not be easily guessed (for example, a favourite pet's name is not suitable)
- be changed every month.

Some network policies force users to change their passwords on a regular basis, and do not allow passwords to be repeated.

Password protection can also be placed on individual files to protect them against being viewed by unauthorised people, as shown in Figure 1.9.

FIGURE 1.9

Password protection has been placed on the file, preventing access to it unless a password is entered.



Selina Dennis

Firewall

Intrusion prevention and detection systems are designed to protect computer networks by effectively identifying and responding to known attack profiles. A **firewall** is one such mechanism to control incoming and outgoing network traffic. A firewall is a server and software combination that filters the information coming through an internet connection into an organisation's internal network. Any packet of data that is flagged by the filters as unwanted is not allowed through.

The filters used by a firewall may include examining the IP address of computers that request information from an internal server, blocking all access to certain domain names, banning certain protocols (e.g. file transfer protocol, mail protocol or Telnet protocol) from accessing particular servers, and searching for certain words and phrases included in packets of information, and blocking transfers that contain them.

A firewall can also be used to restrict employees' access to sensitive information. For example, a firewall can be used to stop some personnel from accessing the payroll database.

Most firewalls use two separate network interface controllers (NICs); one is connected to the internal network and the other to the outside world. Material can only move from one card to the other through the CPU of the server computer that is acting as the firewall. While the data or information is being checked for authenticity, it is also examined for viruses and other malicious codes. Everything that comes in from outside is examined for danger.

Malware protection

Malware is malicious software that includes spyware, adware, Trojan horses, worms and viruses. Spyware and adware use cookies to track the internet sites that a user might visit. Trojan horses can leave your computer open to others to read your personal information by creating back-door access to your system. Viruses and worms can hijack your system to send multiple emails to others or perform other acts of mischief. Both can use up essential system resources, which may result in the computer freezing.

Network administrators usually require workstations to run virus protection software. The antivirus software is often updated automatically via the network. A firewall is also useful to block malware from sending personal information over the internet. Anti-adware programs should also be run on workstations.

Many cyber attacks start with a phishing or spoofing email that attempts to obtain users' personal information.

Ethical issues

Everyone wants the benefits of digital systems; however, intended and unintended negative effects can impinge upon people's rights. As a result, those who design, control and use digital systems have a responsibility to consider the real and potential negative effects and to eliminate or lessen them as much as possible. Sometimes, even this may not be enough to justify the proposed collection, creation or storage of data and information. It is important to take into account legal objections and ethical considerations when creating, acquiring and storing data and information. The purpose for collection needs to be clear (APP 6). This also needs to be articulated in the participant information statements and the consent forms provided to the people from whom information will be sought. Questions of storage also need to be resolved so that data and information cannot be accessed by unauthorised users, and to specify the period of time for which it will be held (APP 11).

Ethics

Ethics are the accepted moral standards that govern the ways in which we behave. These standards may be common across a particular society or specific to a single organisation. They apply to activities that are questionable even if they are not illegal and are distinct from laws. Ethics often provide us with a set of guidelines for appropriate behaviour. If we choose to ignore these guidelines, we may not be committing a crime, but we may be sacked by an employer or shunned by society. For example, the impact of violent video games on children has long been debated. Some people have voiced their concerns that video game writers should not include animated violence in their games because it has a negative impact on children. A system of classification exists for games, similar to television and film classifications, but there is no legal restriction preventing these games from being created.

These examples demonstrate how ethics hinge on society's values and standards. In the video game example, there are two competing principles. On the one hand, some would argue that protecting children from video games that might be harmful is the right thing to do. Others would argue that it is more important to maintain freedom of expression. Often, then, questions of ethics become debates over which of two principles is more important. Such conflict may be said to constitute a **dilemma**. This is especially the case when the consequences of action are open to debate or interpretation.

The standards or guidelines that determine whether an action is good or bad are what we call ethics. Ethics govern, among other things, the use of data collection. Ethical principles or guidelines often have an accompanying law, but the ethical principle is usually broader and the law applies only to certain circumstances or certain applications of the principle.

For example, is it ethical that permission to publish photos of people be gained before promotional material is distributed? Legally, as your image is often not copyrighted, most online images may be used for promotional purposes. Many social media websites are scraped for images, which are then used without permission. Is this ethical?

Transparency

To manage data ethically, after data has been collected, it is important that how it is then used has been made clear. Transparency in this regard involves specifying the original purpose of data collection, how the data is stored and where, who has access, how long the data and information would be kept, and the disposal process. Ownership and who holds copyright must be made clear, prior to collection. Misuse of data and personal information arises when these ethical guidelines are not followed. There are many examples of unethical behaviour online that are legal, including the use of unauthorised celebrity photos or voice characteristics to sell products. For example:

- A well-known betting company has used the recognisable voice of a comedian imitated by a voice actor to promote its betting services. It is legal, but is it ethical?
- Celebrities have had their images associated with online advertising. The people involved have been informed by the courts that their image 'is not copyrighted'. The advertisement does not state that the person endorses the product. The association of the image on the advertisement suggests endorsement, but under the law, the link is not explicit. Is this ethical?

Use of inaccurate or incomplete data

The use of inaccurate or incomplete data in the context of data management and communication raises significant ethical concerns, particularly regarding fairness, privacy and the potential for harm.

Accuracy and fairness

Inaccurate or incomplete data can lead to biased or unfair outcomes, especially in decision-making processes that rely heavily on data analysis. For example, in the context of AI and machine learning, if the training data is biased or incomplete, the resulting models may produce skewed results that unfairly disadvantage certain groups. This can manifest in areas such as hiring practices, loan approvals or the legal system, where biased algorithms may reinforce existing inequalities.

Privacy and confidentiality

Managing and communicating data often involves handling sensitive information. Incomplete data can lead to breaches of privacy if it is not appropriately de-identified or if context is lost that otherwise protects individuals' identities. For instance, partial data sets may leave out important information that ensures privacy, leading to inadvertent disclosure of personal details. This not only violates individuals' rights to privacy but also undermines trust in data-handling practices.

Harm and misinformation

Inaccurate data can lead to harmful decisions and spread misinformation. For example, in health care, using incomplete data for medical research or patient treatment plans can result in misdiagnosis or inappropriate treatments, directly impacting patients' health. In government and legal domains, relying on inaccurate data can lead to ineffective or harmful policies that do not address the real issues, potentially exacerbating societal problems.

Ethical responsibility

Ethically, organisations and individuals responsible for managing and communicating data have a duty to ensure the data's accuracy and completeness. This includes implementing data-verification processes, transparent communication about the limitations of the data and continuous monitoring for biases.

Ownership and control of data

Ownership and control of data are central to ethical issues in the management and communication of data and information. These concerns revolve around who has the right to access, use and benefit from data, as well as who is responsible for ensuring its accuracy, privacy and security.

Because of privacy laws such as the *Privacy Act 1988*, it is necessary to seek permission to collect any data or information that involves people. For example, you should obtain permission (consent) to photograph or video individuals or groups. The organisation or individual seeking permission needs to let the people photographed or videoed know the purpose of the photographs or video and what they may be used for. Permission is usually provided in written form and is often referred to as 'consent'. Permission must be sought because a photograph or video image in which an individual can be identified is considered to be personal information. Pictures of people can be used in advertisements, or for marketing purposes, and sometimes pictures can be used thoughtlessly and depict people in a false light.

Participants in any research need to be informed about what the research entails. They need to know what they are required to do and how much of their time it will take, and how often they will be required. All details of the research need to be given so that participants can make an informed decision to participate. For example, when researchers want to conduct a survey, they need to specify how much time it will take (for example, 15 minutes) and how many times the survey will need to be completed. Participants need to know whether the survey will be paper-based or electronic, and how they will get access to it. All these details need to be explicitly stated so that each participant has a clear understanding before agreeing to take part. The researchers cannot put pressure on the participants or use coercion or



Weblink
Australian
Code for the
Responsible
Conduct of
Research

provide financial or other incentives for them to participate. Participation in research needs to be voluntary and informed.

In universities, research that involves people or animals cannot begin until researchers obtain ethics clearance. Each university has an ethics committee. These committees follow the guidelines set out by the National Health and Medical Research Council (NHMRC) to ensure research is carried out ethically and with integrity. In order to obtain clearance from the ethics committee, the researcher/s must demonstrate that they have followed correct procedures and processes before collecting data to ensure that all risks have been addressed, and that it is established that participation is informed and voluntary.

The *Australian Code for the Responsible Conduct of Research, 2018* (the 2018 Code) ‘establishes a framework for responsible research conduct that provides a foundation for high-quality research, credibility and community trust in the research endeavour’ (NHMRC, 2018, p. 1). This code sets out good research practices that researchers in Australian universities must adhere to. These practices cover important principles such as honesty, fairness, transparency, respect and accountability.



FIGURE 1.10

Consent forms are one method to obtain permission from participants.

Consent forms

One method of obtaining permission for research purposes is to use consent forms. The **informed consent** of all research participants is required. This means that before agreeing to participate in research, they are aware of what the research involves, the time required from them and any risks that may arise. Participation in research not only has to be informed; it must also be voluntary – that is, participants are not pressured to be involved, and have the capacity to make their own decisions based on their understanding of the research.

You must obtain consent when interviewing or observing participants, creating surveys or collecting any type of data.

Consent forms should have the following information clearly listed:

- the title of the project
- the name of the researcher
- what the project is about and why it is being undertaken
- what is required from the participants in terms of time, effort, resources and costs
- the rights and interests of the participants – that they freely consent to be involved in the research and can withdraw at any time without having to provide a reason
- a statement of whether the participant's identity will be preserved
- statements of confirmation, such as:
 - » 'I have been informed of and understand the purposes of the study.'
 - » 'I have been given an opportunity to ask questions.'
 - » 'I understand I can withdraw at any time without prejudice.'
 - » 'Any information that might potentially identify me will not be used in published material.'
 - » 'I agree to participate in the study as outlined to me.'

The form should also have space for the participant's name, their signature and the date.

Name of school:

Project title:

Investigator(s):

- 1 I consent to participate in the project named above. I have been provided a copy of the project consent information statement to which this consent form relates and any questions I have asked have been answered to my satisfaction.
- 2 In relation to this project, please circle your response to the following:

• I agree to be interviewed by the researcher	Yes	No
• I agree to allow the interview to be recorded by an electronic device	Yes	No
• I agree to make myself available for further information if required	Yes	No
• I agree to complete surveys asking me about [Insert topic]	Yes	No
- 3 I acknowledge that:
 - a my participation is voluntary and that I am free to withdraw from the project at any time without explanation;
 - b the project is for the purpose of research and not for profit;
 - c any identifiable information about me which is gathered in the course of and as the result of my participating in this project will be (i) collected and retained for the purpose of this project and (ii) accessed and analysed by the researcher(s) for the purpose of conducting this project;
 - d my anonymity is preserved and I will not be identified in publications or otherwise without my express written consent.

By signing this document I agree to participate in this project.

Participant name:

Signature and date:

FIGURE 1.11

Sample consent form

Participation information statements

The most common way to obtain informed consent is to provide a **participant information statement**, written in plain English, along with a consent form. This provides participants with information about the research and also provides the scope to answer questions that participants may have. Once they have read and understood the information statement, they are asked to sign the consent form to indicate their agreement to participate. When developing a participant information statement, consider the following.

- Use language your audience can easily understand. The statement should be suitable for a 12-year-old reading age. If technical terms are required, make sure they are clearly explained the first time they are used.
- Write in a conversational style, as if you were speaking to the participant.
- Language used should be clear, concise, invitational, culturally appropriate and logically set out.
- Use pronouns, such as 'I', 'we' and 'you'. This encourages the use of active voice and will be clearer to the reader. For example: 'You will be asked to participate'.
- Use reader-friendly formatting so that your document is easy to read.
- Ask others to read and edit your document.
- Include a statement on how the data will be stored.
- State where and how the research findings will be published.
- Give further information about the project, such as whom to contact.

Even if the data-collection technique is an online survey and the researchers do not meet the participants, respondents must still be informed about the research, and asked to consent online before completing the survey. They may be asked to consent by accepting and selecting the 'I agree to participate' option. If they choose the 'I do not wish to participate' option, then they do not get access to the survey. See Figure 1.12 for an example of an integrated online information statement and consent form.

Technology Disconnect

The transition from secondary school to university is filled with expectations that may or may not be met once you start your tertiary studies. In particular, expectations with respect to the use of technologies for learning and teaching - both by teachers/lecturers and you as a student - is one of the key areas of interest for many universities. Please be assured that your responses will be kept completely confidential. This project aims to discover what your expectations were for technology use at university while you were still in secondary school, and compare that to the reality you have encountered since you started your university studies. We would like to invite you to tell us what you thought you would find, and what you have actually found to be the use of technology for learning and teaching at university. The study is a voluntary, anonymous, online survey that should take you no longer than 15 minutes to complete. While we encourage you to complete the whole survey, you may stop at any time and only those responses you have already answered will be used in the study.

The anonymous findings will be collated before being used:

- to set strategic directions around the use of technology for learning and teaching
- to inform future investment in technologies for learning and teaching
- to inform professional development for staff in the use of technologies

It is critical to have the student voice guiding the above so we strongly encourage you to participate. If you would like further information about any aspect of the project before agreeing to participate, please do not hesitate to contact us:

██████████

Associate Professor (██████████)

██████████

██████████

Any queries about your participation in this project may be directed to the Primary Investigators listed above. If you have any concerns or complaints about the conduct of this project, please contact:

Research Ethics Officer, ██████████, ██████████

F ██████████

By clicking the button below, you acknowledge that your participation in the study is voluntary, you are 18 years of age, and that you are aware that you may choose to terminate your participation in the study at any time and for any reason.

Please note that this survey will be best displayed on a laptop or desktop computer. Some features may be less compatible for use on a mobile device.

I consent, begin the study

I do not consent, I do not wish to participate

FIGURE 1.12

An example of an integrated online information statement and consent form. Note that personal information has been redacted for privacy reasons.

Source: Google LLC

The consent information statement should be clearly labelled as such (or quite similar) and contain key pieces of information as follows:

Project title

Give the project a full title.

Investigators and other project personnel

List clearly all researchers.

Introduction to project and invitation to participate

Both introduce the project and invite participation.

What this project is about and why it is being undertaken

Sufficiently explain what the project is about, its aims, and why it is being conducted.

Project and researcher interests

Researcher and project interests should be sufficiently disclosed as applicable; for example, project is partly, mainly or wholly to satisfy the requirements for a student's VCE subject.

What participation will involve – time, effort, resources, costs

Give sufficient clear detail as to what is being asked of participants – voluntary consent to their time, effort, supply of information/body tissue/records/personal effects, etc. Avoid language that can be read as orders or directives (i.e. not 'You will do this or that'; but better as 'We will ask you', etc.) and presumptuous language (e.g. 'Dear Participant').

Participant rights and interests – risks and benefits/contingencies/back-up support

Outline realistically any potential risks (minimal or otherwise) and what preventative, minimisation or redress arrangements are in place. If some research questions or issues can be considered particularly sensitive, give sample questions or topics as an indication of the information that will be discussed or requested. Describe any benefits pertaining to individual participation or more generally. Avoid grandiose claims.

Participant rights and interests – free consent/withdrawal from participation

Participation should be voluntary, and free from any coercion or perceived coercion. Detail on this matter should be clear – that an individual is free to participate or not, and the circumstances. If, for example, the participants are students, patients or employees, it will help to clarify that their decision whether or not to participate will have no bearing on their results, treatment or employment (in some cases this may need further explanation, such as details about recruitment).

A statement about the participant's right to withdraw participation, data or material contributed, ordinarily without question or explanation, needs to be included.

Don't forget to outline how valid consent is to be obtained (by signed consent form, completion and return of an anonymous survey, any witnessing procedure if applicable, etc.). Often it helps to highlight or bold this part.

Participant rights and interests – privacy and confidentiality

Give clear information about secure arrangements for data access, collection, use, retention and/or disposal. This needs to comply with mandatory Australian Privacy Principles.

If signed consent forms are required, state whether they will be stored separately from any data collected and who will have access to them.

Remember, people are increasingly concerned about data access and data matching. Clear information will help allay any concerns.

Research output

Outline intended or anticipated publication or reporting of research findings. If need be, reiterate or refer to privacy arrangements for confidentiality/anonymity. Offer to make available any report or article or summary, where appropriate, and indicate how this will occur.

Further information about the project – whom to contact

You will need to nominate at least one person to contact regarding further information about the research activity or participation in the project.

FIGURE 1.13 Information that should be included in a participant information statement

Misuse of personal data and information

Misuse of personal data and information presents significant ethical issues in data management and communication. Data and information misuse raises ethical issues by compromising informed consent and fostering discrimination and profiling. When personal data is collected or used without individuals' explicit consent, it breaches ethical standards and erodes trust. Additionally, using personal data for decision-making in contexts like hiring,

lending or law enforcement can lead to unfair treatment and systemic biases, as data misuse often perpetuates discrimination based on race, gender and/or socioeconomic status. This means that strong data protection measures and transparent practices must be put in place to uphold the integrity and fairness of data management and communication.

Repurposing and sharing of data by artificial intelligence systems

Repurposing and sharing of data by AI systems present significant ethical issues in data management and communication. When AI systems use data beyond its original intent without clear consent, it can lead to privacy violations and a loss of trust. Additionally, sharing data across different platforms or with third parties increases the risk of misuse and unauthorised access. Such practices can result in biased outcomes if the data is applied in new contexts without considering its limitations and relevance. Ensuring transparency about data usage, maintaining robust consent processes and implementing stringent data protection measures are essential to address these ethical concerns.

Privacy issues and AI systems

Imagine a health app that collects users' personal health data to provide personalised fitness recommendations. Without explicit consent, the company decides to repurpose this data for developing a new AI-driven health insurance risk assessment tool. Additionally, the data is shared with third-party insurance companies to refine their pricing models.

The repurposing and sharing raise several ethical issues. Users originally provided their data for fitness advice, not for determining insurance premiums. The lack of informed consent breaches ethical standards, potentially leading to privacy violations and eroding user trust. Furthermore, sharing the data with insurance companies risks misuse and unauthorised access, which could result in unfair discrimination against users based on their health data.

This scenario highlights the need for transparency, robust consent processes and strict data protection measures to ensure ethical data management and communication when repurposing and sharing data by AI systems.

CASE STUDY



Handling ethical issues

The six steps outlined below for handling ethical dilemmas act as a framework that can be used within an organisation to solve legal, ethical or social tensions. This framework will provide support and guidance in making a decision when presented with a dilemma. Once the organisation has solved its problem, it may wish to update any policies concerned to ensure that the process for resolution is clearer.

- 1 Identify the problem: What decision has to be made and what facts are required?
- 2 Identify the stakeholders: Who are they? What interests do they have? What power do they have? Who is vulnerable? How are the vulnerable to be protected?
- 3 Identify possible alternatives: What options are available? What are the likely consequences?
- 4 Identify ethical standards: Are there any applicable laws? Are there any morals or standards that could be applied? Is there a precedent?
- 5 Evaluate options: Identify strengths and weaknesses. Identify the option that causes least harm. Can the decision be reversed?
- 6 Make a decision: Select the preferred option. Justify the option. Communicate the result of the decision to all stakeholders.

1

CHAPTER SUMMARY

Essential terms

accurate correct in all details and free from errors

backup the process of copying files from an information system to some type of storage device to guard against possible data loss

bias prejudicial or unreasoned judgement

Boolean data type a logical data type; Boolean data can hold one of only two possible values, usually true or false

character data type a data type representing a single letter, number, symbol, punctuation mark or space

closed (or closed-ended) questions queries that are restricted in the range of options provided so that only specific answers are elicited

data unprocessed, unorganised and distinct facts or ideas; in addition to text and numbers, data also includes sounds, images and video

data integrity the quality of the data

data types different forms that variables and data may take, that determine the data that a variable may contain, and how the data or variable may be manipulated

decrypt to translate encrypted data back into ordinary text that can be read by anyone

de-identify to remove information from data so an individual cannot be identified

differential backup used in conjunction with a full backup; only files that have been altered since the last full backup are copied; restoration requires the full backup to be restored first, followed by files from the differential backup

dilemma when people must choose between two (or more) equally desirable (or undesirable) options; for example, between allowing the sale of violent video games to preserve freedom of expression and banning their sale in order to protect children from possible harm

encryption the process of encoding or changing data so that an unauthorised user who reads the data would not be able to understand it

ethics accepted moral standards that guide behaviour; these standards may be common across a particular society or specific to a single organisation, and they apply to questionable activities over and above any legal requirements; ethics often provide us with a set of guidelines for appropriate behaviour

exponent the power of 10 by which the mantissa in a floating-point number must be multiplied to regain the original number

firewall hardware and software that restrict access to data and information on a network

floating-point number a number with a fractional or decimal component

footnote a reference that is listed at the bottom of the page on which the citation is made

full backup copying all chosen files to a backup device; it can be slow to perform, but is the easiest and quickest form of backup from which to restore data

hypothesis testable statements that are based on probabilities about what will happen according to the applied theory

incremental backup similar to a differential backup in that it works in conjunction with a full backup, and only backs up files that have been altered since the last incremental backup, but it uses more than two backup media; it is the most complicated strategy from which to restore files

information processed, organised and value-added data, which can be paper-based (hard copy) or digital (soft copy)

informed consent a necessity for all participants before agreeing to take part in research; participants must be informed of what the research involves, the time commitment expected and any possible risks that may arise

integer a number without a fractional or decimal component

interview a conversation, usually between two people, in which questions are asked and answers are given

malware short for 'malicious software'; programs designed to infiltrate and cause damage, disruption or access to a device or network without the user's knowledge or consent; includes viruses, worms, Trojan horses, adware, spyware, logic bombs and keyloggers

mantissa all of the digits in a floating-point number, with a decimal point after the first digit that is not zero

open-ended questions queries that allow people to answer in the manner they wish

participant information statement a document that provides participants with information about the research in an unbiased way, and also provides the scope to answer questions that the participant may have

password a secret term used to identify the user

plagiarism passing off someone else's work as your own

precision being exact

primary sources original, unprocessed data and resources; that is, information that has not been processed, analysed or interpreted in any way, such as interviews, speeches, emails, debates and meetings; primary data usually comes from stakeholders

pseudonym a fictitious name that is given to a person, or that is chosen by a person, to hide or protect their identity

qualitative data collected data that is based on subjective data-collection techniques such as interviews, focus groups, video footage and observation

quantitative data collected data that is measurable and specific; quantitative data gathering is based on verifying a theory through the use of statistics and data that is largely numerical

referencing citations in a document that assist readers to know where to locate the source of an original idea or quote in a piece of work, and assist students to avoid plagiarism

relevant appropriate to the discussion

reliable able to be trusted

research question a question that enables a researcher to narrow the focus of the topic of the investigation

secondary sources sources of information that has been processed, interpreted or analysed in some way by other people, such as textbooks, websites, magazines, newspapers and TV programs

stakeholder an individual or group who either has an interest in, or is affected by, the decisions and actions of an organisation

string a data type representing a set or sequence (collection) of characters

survey usually a set of questions that ask for a response from a list of alternatives, such as A, B, C, D, or from a range, such as 1 to 5 or very low to very high; surveys can easily be given to many people, and are easily processed and analysed using computer-based methods because the answers can be recorded as numbers

theory a general statement that describes something, provides an explanation of why something happens and can be applied to predict what will happen in the future

unencrypted data data that is not protected by encryption and can thus be read by anyone; also known as 'plain text'

username the name given to the user on a computer or computer network

variable in programming, a key word, phrase or symbol that represents a value that may change

vested interest arises when an individual, group or organisation has a strong personal interest because there is an advantage to be gained

Important facts

- 1 Artificial intelligence (AI) has developed through a new type of neural network called large language model (LLM). A generative pre-trained transformer (GPT) samples many hundreds of millions of examples and establishes patterns and trends using hundreds of parameters. ChatGPT is a chatbot interface that attempts to provide human-like responses to input prompts. Each input prompt generates a different response, even from identical input prompts. Any copyright is held by the person inputting the prompt.
- 2 Data must be relevant to produce usable information. Data needs to be processed while it is current because decision-making should not be based on outdated data.
- 3 Data that is entered into a computer must be accurate. Transcription is often a cause of error. Transcription errors occur when the person entering the data misreads the information through, for example, a lapse in concentration or being interrupted, or presses the wrong key.
- 4 Interviews are usually done one-to-one, but can sometimes be done in groups, and can take a substantial amount of time. A major feature of an interview is the opportunity for in-depth follow-up and clarification questions that cannot be done with surveys, which are often answered in private. Interviews are very useful for eliciting the feelings, attitudes and opinions of people that are too complex to easily record in a survey.
- 5 Bias can infiltrate data if the respondent to a survey or interview has a vested interest in the outcome of the research, if the timing of the data gathering is inappropriate, or if the chosen sample size is too small.
- 6 Timing of events needs consideration when collecting data as it can cause skewed results, which can lead to inaccurate or misleading conclusions.
- 7 Sample size must relate to the purpose of the data collection. Generally, a larger sample size leads to greater precision.
- 8 There is a plethora of unchecked information on the internet; however, some of the views presented may not be widely accepted or proven. Sources cited should be reliable.
- 9 The American Psychological Association (APA) created a style guide to assist with academic writing such as essays, books and other publications. The APA style is widely used and is the reference style that students are expected to use.
- 10 Privacy is a fine balance between the interests of researchers and those of the participants. Privacy laws attempt to stop inappropriate intrusion into the lives of individuals. Often, however, the problem is not the collection of data, but how the data is used or misused by people entrusted with it.
- 11 Researchers must ensure that data and materials generated and collected as part of their research, regardless of the format, are stored securely in a durable and accessible form. Stored data can be protected with both physical and software-based controls, such as backing up of data and shredding of confidential documents.
- 12 Cloud-computing companies provide off-site storage, processing and computer resources to individuals and organisations. These companies are typically third party and they store data to a remote database in real time.
- 13 To verify users' rights to access a network, security features are required. A system of establishing usernames (or user IDs) and passwords allows for the identification and authentication of each user.
- 14 The *Privacy Act 1988* (Cwlth) was amended by the *Privacy Amendment (Enhancing Privacy Protection) Act 2012*. This came into effect in 2014. As part of this Act, the Australian Privacy Principles (APPs) replaced the National Privacy Principles and the Information Privacy Principles so Australia now has one set of privacy principles. The APPs apply to Australian government agencies.
- 15 APP 2, 'Anonymity and pseudonymity', offers individuals dealing with organisations the option of using a different name or a pseudonym in relation to a particular matter.
- 16 APP 6, 'Use or disclosure of personal information', states that the information that is being held is in line with the primary purpose it was intended for. Information cannot be used for a secondary purpose without consent from the individuals concerned.
- 17 APP 11, 'Security of personal information', states that reasonable steps need to be taken to protect personal information an organisation holds from misuse, interference and loss, and from unauthorised access, modification or disclosure. The organisation also needs to destroy or de-identify personal information in certain circumstances.



Research, data and backups

- 1 What can occur if information is produced from incorrect or incomplete data?
- 2 Why is it important to ensure that data is accurate?
- 3 What are the properties of usable data?
- 4 What is a common cause of inaccurate data?
- 5 How can the accuracy of a primary source be determined?
- 6 With an example, explain the importance of timeliness in ensuring the quality and usability of data.
- 7 What influences the introduction of bias into data?
- 8 What is the difference between quantitative and qualitative data? Provide two examples.
- 9 What strategies could be used when gathering quantitative data?
- 10 Provide an example of referencing based on the APA style.
- 11 Why is it important to obtain permission when collecting data?
- 12 What is the purpose of consent forms?
- 13 Why is encryption important in data security?
- 14 How do usernames and passwords protect data?
- 15 What makes a strong password? Provide an example of a very strong password.
- 16 Describe a strategy for backing up data.
- 17 What is the difference between a full backup and an incremental backup?
- 18 Why is it important to secure data when conducting research?
- 19 How can the use of AI-GPT assist with administering surveys or interviews?
- 20 What are some implications of using AI-GPT to summarise surveys and interviews?

Legal and ethical Implications

- 21 How do the Australian Privacy Principles affect the individual?
- 22 Under which legislation do the Australian Privacy Principles fall?
- 23 Why is it important to de-identify personal data?
- 24 What is an ethical dilemma in the context of data collection for research purposes?



APPLY YOUR KNOWLEDGE

Street traffic

- 1 Milorad believes that there is too much traffic in his street, particularly on weekdays – in the morning and evening. Milorad lives in a small residential street, but cars use this street as a shortcut to avoid traffic lights and the main roads. Milorad wants to do some research to support his theory about the traffic on his street, and present a case to the local council.
 - a Clearly state the topic Milorad will investigate.
 - b What type of data will Milorad need to collect to assist in his investigation?
 - c Identify an appropriate data-gathering technique Milorad could use.
 - d Justify the selected data-gathering technique.
 - e How will Milorad keep the data safe?
 - f Does Milorad need to get permission to conduct his research and, if so, from whom?
 - g What tools will Milorad use to interpret the results?
 - h What types of relationships and patterns is Milorad looking for?
 - i How will Milorad present the data to the local council?



Internet usage

- 2 Go to the UN Data website and select 'Datamarts'. Then find the World Telecommunication/ ICT Indicators database from the list of all the data sets provided. Select to view the data on the percentage of individuals using the internet.
 - a Filter the data to show only the value of countries using the internet in 2010–14 from 10 countries. It is best to select from a mixture of countries to highlight the differences between them in levels of access to the internet.
 - b Copy and paste this data into a spreadsheet program.
 - c Create a column chart and a scatter diagram chart to depict these statistics.
 - d Discuss which graphic representation best conveys the data and why.

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

- KK1.1.8: characteristics of functional and non-functional requirements, constraints and scope
- KK1.1.9: design tools for representing the functionality and appearance of databases, spreadsheets and data visualisations
- KK1.1.10: structural characteristics of relational database management systems (RDBMS)
- KK1.1.11: structural characteristics of spreadsheets
- KK1.1.12: software functions and techniques for efficiently and effectively manipulating, validating and testing data to develop databases, spreadsheets and data visualisations
- KK1.1.13: use of spreadsheets to calculate descriptive statistics for analysis
- KK1.1.14: purposes of data visualisations for educating, entertaining, informing and persuading audiences
- KK1.1.15: types of data visualisations
- KK1.1.16: components of data visualisations
- KK1.1.17: formats and conventions suitable for databases, spreadsheets and data visualisations.

Victorian Curriculum and Assessment Authority, Victorian Certificate of Education Applied Computing Study Design 2025–2028, VCAA, 2024, pp. 25–6

FOR THE STUDENT

If you can imagine the amount of data that is generated every day just by using social-networking tools, you might also be able to imagine that there is someone, somewhere, who is looking through a mountain of data for meaning. Data visualisation is the process by which we take large amounts of data and process it into effective graphical representations that will meet the needs of users or clients. These representations can take the form of charts, graphs, spatial relationships and network diagrams. In some cases, the data visualisation might involve interactivity and the inclusion of dynamic data that allows the user to deduce further meaning from the visualisation.

This chapter introduces you to the knowledge and skills needed to use software functions and techniques to access authentic data from repositories and present the information in a visual form.

The key knowledge and skills are based on Unit 1, Area of Study 1. If a data visualisation is effective, it reduces the effort needed by readers to interpret the information. This chapter takes you through the different types of visualisations, and then uses a case study to explore some of the tools available to process data from various sources and repositories, such as the domestic energy consumption records, the Bureau of Meteorology and the Australian Bureau of Statistics.



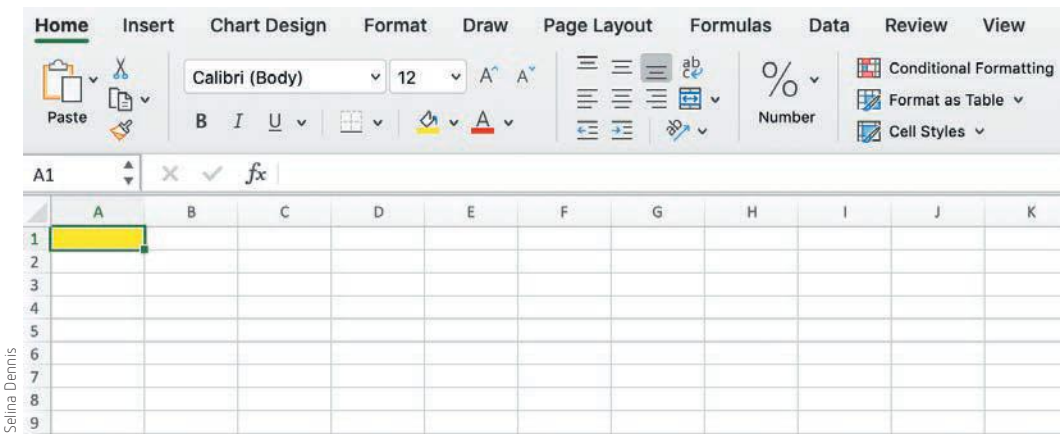


FIGURE 2.3 Excel spreadsheet layout. For example, A1 is the first top left cell.

Cell naming

In Excel, a name can be given to a cell or range of cells. For example, instead of using the cell reference A1 or A1:C1, simply use the name you have assigned to it. When referencing those cells in a formula, just use the assigned name: =SUM(VALUEs) instead of =SUM(A1:C1).

TABLE 2.2 Summary of reference types when a formula containing a cell reference is copied one cell down and two cells to the right (from A1 to C2)

If the reference is:	It changes to:
\$A\$1 Absolute column Absolute row	\$A\$1 The reference is absolute
A\$1 Relative column Absolute row	C\$1 The reference is mixed
\$A1 Absolute column Relative row	\$A3 The reference is mixed
A1 Relative column Relative row	C1 The reference is relative

There are several methods for creating name ranges in Excel.

Method 1: Use Define Name

Select by highlighting the range you want to name. Go to Formulas > Define Name.

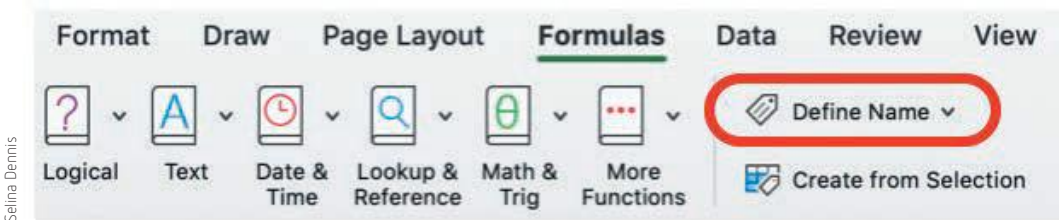


FIGURE 2.5 Create a named cell range under Formulas > Define Name.

Method 2: Use the Name Box

Select by highlighting the range of cells that you want to name. Go to the Name Box at the left of the Formula bar, then type the name you have chosen for those cells. Note that the

2.1

THINK ABOUT APPLIED COMPUTING

Search the web for information about the advantages of using R1C1 and A1 notation. Consider:

- relative references
- formulas in tables.



FIGURE 2.4 Relative positions of cells A2 and C2

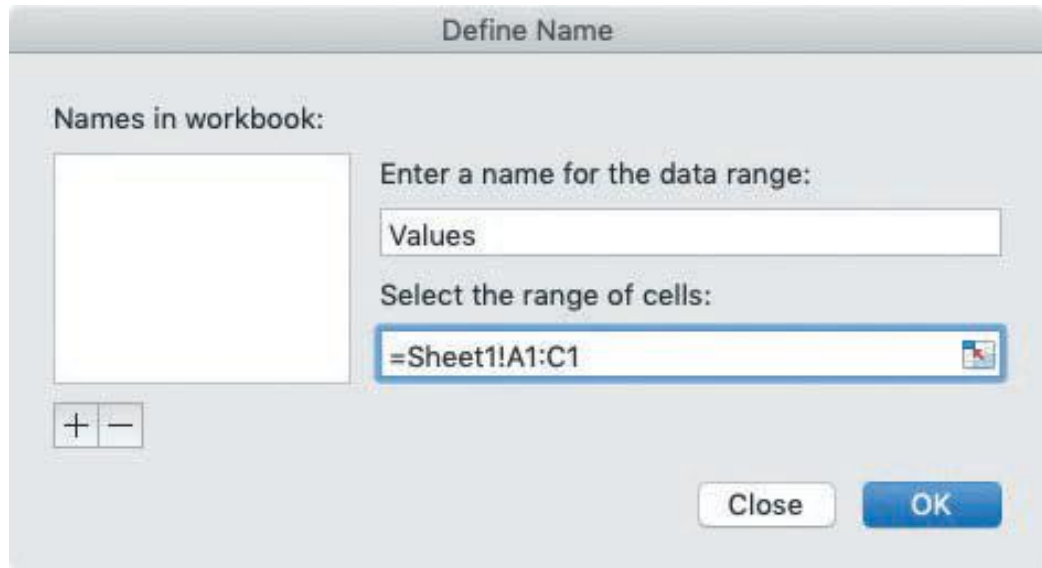


FIGURE 2.6 Enter the chosen name and range.

FIGURE 2.7 Create a named range by typing directly into the Name Box.



name created by Method 2 will apply to the entire Workbook. If you want a local name that only applies to this worksheet, use Method 1.

Method 3: Use Create from Selection

Go to Formulas > Create from Selection. Then select one or more of the options.

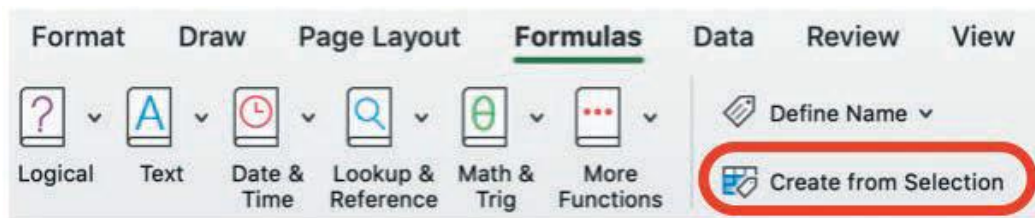


FIGURE 2.8 Create a named cell range under Formulas > Create from Selection.



FIGURE 2.9 This method is useful when you have Column Headers. If you choose several boxes, then several names will be chosen from the field names.

A spreadsheet is not a database

Many of the functions of a spreadsheet can achieve similar results to a database. For example, each spreadsheet row behaves like a record; data and records can be sorted, searched and filtered; and data can be extracted from other sources. The main difference is the size of the data set that can be manipulated. Spreadsheets require the data set to be held in the RAM of the local computer. For Excel, this is a limit of around several million records or approximately 10 GB, depending on the memory specifications of the computer. Database applications have no limit to the size of the files that are accessed. The efficiency of the operation – for example, search or sort – will be affected by the complexity of the instruction and the number of records to be processed. Processing times would be measured in hours or days in these instances.

In many business operations, where the file size is small and the number of records is only a few hundred thousand, spreadsheets are used instead of dedicated database applications. The opportunity for error is greatly increased if cells are left 'open' or inexperienced users gain access to the master file.

Databases

A database is an organised collection of structured information or data, designed to manage, retrieve and manipulate data efficiently.

Tables

In databases, tables are similar to worksheets in spreadsheets. Each table stores data about a specific entity. For example, a 'Customers' table in a database would be similar to a 'Customers' worksheet in a spreadsheet. The table could contain rows for each customer and columns for customer details such as name, address and phone number.

Columns and data types

Columns in a database table are comparable to columns in a spreadsheet but with enforced data types. For instance, in a database, a 'BirthDate' column in a 'Customers' table could be defined with a date data type to ensure it only stores date values. In a spreadsheet, a 'BirthDate' column could contain different types of data unless rules were implemented to prevent this, but in a database this consistency is automatically enforced.

Primary keys

Primary keys in databases uniquely identify each record in a table, similar to how you might use a unique identifier in a spreadsheet. For example, an 'EmployeeID' column in an 'Employees' table serves as the primary key, ensuring that each employee record is unique. In a spreadsheet, you might manually assign unique identifiers to each row to achieve a similar effect but a database enforces this automatically.

The image shows a screenshot of a database table design tool. The table is named 'Subjects' and has two columns: 'Field Name' and 'Data Type'. The 'Field Name' column lists various attributes, and the 'Data Type' column specifies the type of data each attribute can hold. A key icon in the first row indicates that 'subjectCode' is the primary key. Annotations with arrows point to the table name, the primary key, the list of fields, and the list of data types.

Field Name	Data Type
subjectCode	Short Text
subjectName	Short Text
yearLevel	Number
wholeYear	Number
folio	Number
lote	Number
faculty	Short Text
required	Number
recommendation	Number
vcevm	Number
music	Number
active	Number

FIGURE 2.10 Table design in a database management system

Foreign keys

Foreign keys in databases establish relationships between tables, similar to how references are used in spreadsheets to link data across different sheets. For example, an 'OrderID' in an 'OrderDetails' table acts as a foreign key that links to an 'OrderID' in an 'Orders' table. This ensures that every order detail corresponds to a valid order. In spreadsheets, you might link data manually across sheets, but databases enforce these relationships automatically to maintain data consistency and integrity.

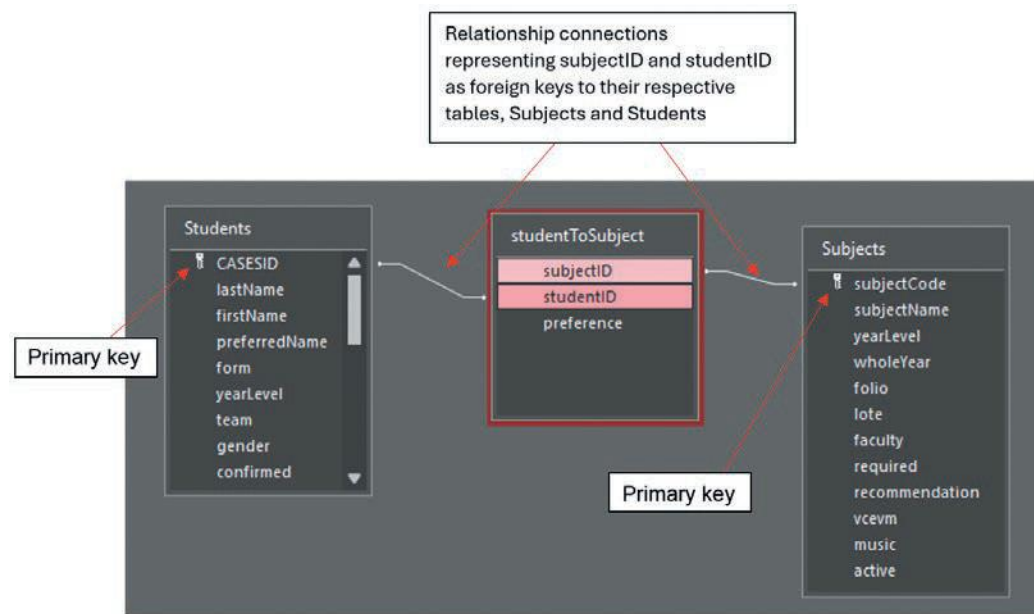


FIGURE 2.11 Table design representing relationships with primary and foreign keys

Formats and conventions for database activities are not well defined. The basic rule is for ease of use and predictability. Every database developer has their own style. The more experience you gain with database operations, the more you can adapt to the style of the database developer. Consistency is the strength of any convention, though exceptions may be permitted, if the reasons are documented and recorded.

Some common conventions for databases are:

- Adopt a pattern and apply the rules consistently to tables, views/reports, columns and keys.
- Camel case is a simple naming convention to apply, where the first word is in lower case and the other words start with upper case – forExampleCamelCaseImprovesReadability.
- Include the type in the name (see Table 2.3).

TABLE 2.3 Naming conventions for database elements

Table	tbl_EnergyUse
Report	rpt_EnergyConsumption
Primary key	date_id
Foreign key	location_id

Naming conventions are a matter of choice. Your task is to consistently apply whatever scheme you decide on, and document what you are doing. When maintenance is required some months later, you may be required to revisit the database scheme. To avoid frustration and wondering what the developer was thinking, a documented naming approach is recommended.

Spreadsheets

The first SSRB standard recommends that a spreadsheet should ‘*always be considered from a user’s perspective rather than a developer’s perspective*’. BPMS1-1 Clarity of purpose must be a primary consideration for the developer when creating workbooks, sheets and components within a sheet.

An overall approach for every spreadsheet is recommended to adopt ‘purpose-based formatting’. In our situation, this will require distinguishing between sheet types and worksheet cell ranges.

The main ideas for spreadsheets include:

- naming sheets with descriptions of purpose
- using fill colour to define areas of sheets
- using font colour to indicate the content of a cell
- using range names to describe the content or use of the range being named (see cell naming on pages 41–2)
- aligning data with the same type consistently down rows and across columns
- using number formats that are right-aligned
- clearly indicating or labelling all units and denominations.

Workbooks should be set to calculate automatically, as pressing the function: F9 key for manual calculations is not widely known.

If you are developing advanced interactive models, then the range naming prefixes will become useful. Less complex spreadsheets will not require as many categories because the number of range names will not be required.

A word of caution about using macros

In many cases, the use of **macros** can greatly improve the capabilities, functionalities and user-friendliness of spreadsheets. However, they should be used with extreme care as they are not forgiving if not recorded or written correctly. Macros are likely to inadvertently cause irreparable damage to a spreadsheet. Backups and saved versions are essential when developing macros.

TABLE 2.4 Generally, it is recommended that every workbook contains a range naming key to prevent any potential confusion in understanding the naming system used in the workbook.

Range Type/purpose	Description/purpose	Prefix
Row Array	Single row, multiple-column, single-area array	RA_
Column Array	Single column, multiple-row, single-area array	CA_
Block Array	Single area, multiple-cell, non-row, non-column array	BA_
Base Cell	Single cell base cell (for INDEX function or OFFSET function reference, etc.)	BC_
Lookups	Names a lookup table range on a lookups sheet	LU_
Hyperlink Cell Reference	Hyperlink cell reference	HL_
Check Box Cell Link	Check box cell link	CB_
Drop Down Box Cell Link	Drop down box cell link	DD_
List Box Cell Link	List box cell link	LB_
Option Button Cell Link	Option button cell link	OB_
Spin Button Cell Link	Spin button cell link	S_
Scroll Bar Cell Link	Scroll bar cell link	SB_

Adapted from ‘BPMC 9-6 Range Naming Prefixes’, www.ssrb.org/resources/standards/ce/sma9/range_naming

In 2003, the **Spreadsheet Standards Review Board (SSRB)** was established to maintain the Best Practice Spreadsheet Modeling Standards (BPMS). The full documentation is available as a free download from the weblink.



Weblink
Best Practice
Spreadsheet
Modeling
Standards

Advanced use of range names allows use of an **INDEX** function to dynamically update the worksheet data. Advanced Excel tutorials are beyond the scope of this textbook; however, you could search ‘Excel INDEX function dynamic Name Range’ on the web to discover details about advanced Excel features.

A macro recorder may be employed to create simple automated execution of a sequence of instructions. No changes can be made to any of the objects (sheet names, range names, charts, etc.) if the macro is to operate reliably. Macros are not recommended for calculations. Macros will be slower than worksheet formulas and not well understood by other developers and users. Macros are often disabled as a cyber security precaution. Macro use is then confined to authorised users with a clear reason to have access to such a powerful tool.

Software functions

Modern software provides developers with many tools to select and shape the data in order to allow display and reporting of chosen trends, highlights and insights. Many of these functions are regularly applied by advanced users as the ‘standard treatment’ of manipulating or ‘wrangling’ data prior to reporting or displaying visualisations. Further processing of data creates summaries and aggregations that can provide historical trends and allow accurate predictions. Many visualisations can be arranged to be dynamically updated. This can occur automatically to provide a minimal time delay between data capture, manipulation and report being displayed. While most data is not time-sensitive or volatile, many financial transactions can be manipulated to advantage if up-to-the-second data analysis is available.

Statistics for analysis

Both spreadsheets (e.g. Excel, Numbers, Calc) and databases (e.g. Access, Base) have in-built functions that can quickly calculate summary information based on the selected data.

Basic mathematical and procedural functions include the following.

Measures of central tendency or average: mean, mode and median

→ As you may recall from junior mathematics, the **arithmetic mean** (common average) is equal to the **sum** (or total) of the values divided by the count of the values.

$$\text{arithmetic mean} = \frac{\Sigma(\text{values})}{n(\text{values})}, \text{ where the upper-case sigma, } \Sigma \text{ is Greek 's' for sum}$$

→ **Mode** is the most frequent value.

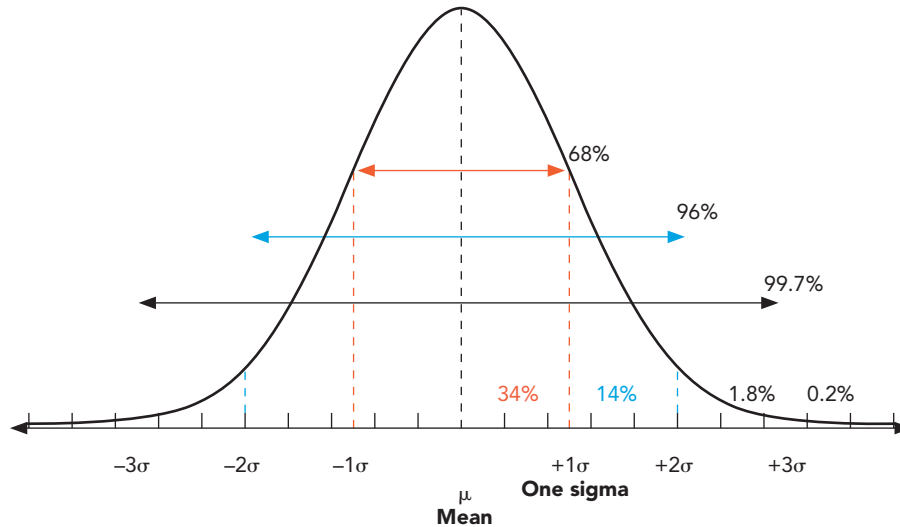
→ **Median** is the middle value, determined after the values are sorted into increasing/decreasing order.

Standard deviation and a ‘normal’ curve

The symbol for **standard deviation (SD)** is a lower-case sigma, σ (Greek letter ‘s’).

The normal distribution or ‘bell’ curve signifies a normal distribution with the average (or arithmetic mean), $\bar{x}$ (pronounced eks bar), also called μ (mu is the Greek letter ‘m’), at the centre. Each standard deviation step from the middle represents a known fraction of the population: $\pm 3\sigma$ encompasses 99.7 per cent of a population, or three in a thousand outliers; $\pm 6\sigma$ (six sigma) encompasses 99.9997 per cent of a population, or three in a million outliers. In Australia, with a population of approximately 30 million, there will be 90 people outside the experience of six sigma of the population. This could apply to height, age or health. Outliers will always occur, so care must be taken removing legitimate data values, otherwise we just describe the ‘average’ or central experience. This is the difference between an average and a normal value.

Note: For a normal distribution, mean, mode and median all have the same value. Despite a 99.9997 per cent probability, there is always the chance there will be instances occurring beyond the six sigma 'limit'. There will always be 'outliers'. The standard deviation, σ , is an indication of dispersion. Wide distributions have a large value of sigma; in contrast, narrow distributions have a small value.



Everything is 'normal'; very little is 'average'. A normal curve describes 100 per cent of the population. Average is only the 50 per cent value.

It is always true that half of every population is above (or below) average.

FIGURE 2.12 Normal distribution always has the same shape. Mean (Greek: μ) and standard deviation (Greek: σ) six sigma covers 99.99966 per cent of the population, or 3.4 outliers per million.

For example, subject scores of 27, 31, 36, 29, 22 can generate mean, mode, median and a standard deviation.

Statistics for MS Excel

Excel has a very long list of in-built functions. Many of these can perform statistical analysis (see Figure 2.13).

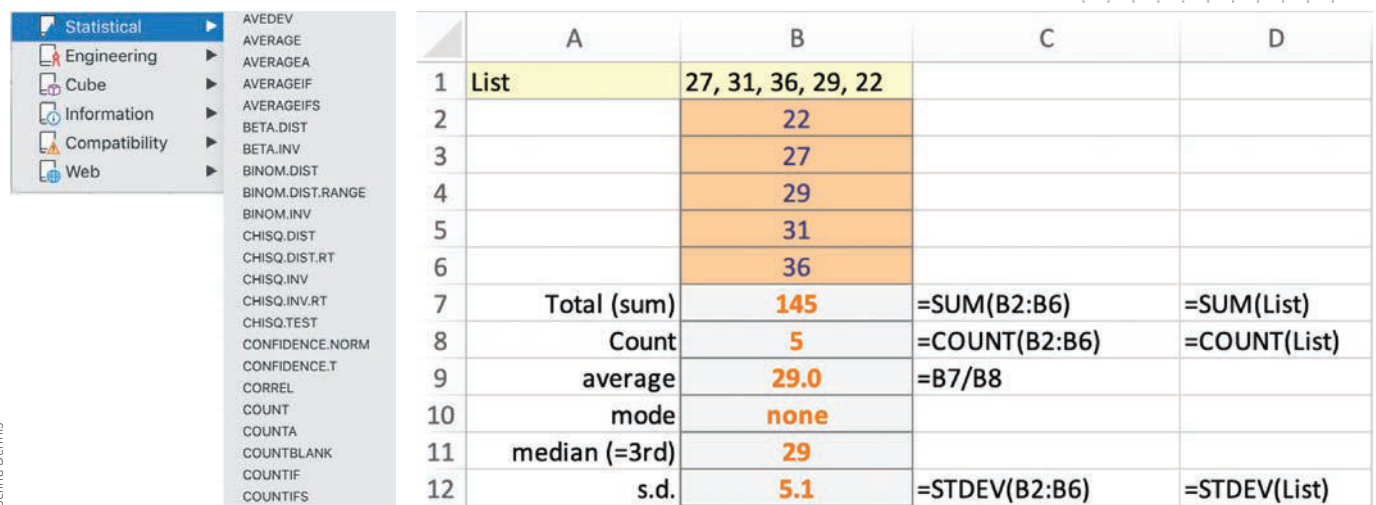


FIGURE 2.13 Example of Excel functions mean, mode, median and standard deviation calculations

Calculations are performed in Column B.

The formulas, as shown in Column C, calculate values.

The average of the list is in Cell B9, where SUM(B2:B6) / COUNT(B2:B6).

When the cell range B2:B6 is referred to many times, it could be named List.

As shown in Column D, **Define Name** is located under Formulas.

When sampling populations, there are several options for mathematical analysis of the data. There are a range of further statistical functions that can describe the confidence level, margin of error and the standard error.

Note: These functions are not within this VCE study design; however, these statistical functions can be used if there is sufficient interest.

Structured Query Language

Structured Query Language (SQL) was initially developed by IBM in the early 1970s and called SEQUEL (Structured English **Q**Uery Language). SQL is now widely used in web development and database management. Anyone having a working knowledge of simple SQL operations is well placed to understand how to establish connections between data and information. However, there are incompatibilities between the numbers of variants that have evolved during the last 50 years.

Often the differences are caused when desirable new features are not backwards compatible and the existing user base places a regard for higher performance and continuity over compliance with SQL standards. For example, DATE and DATETIME functions operate inconsistently between different implementations. SQL code can rarely be imported between database systems without modifications and testing.

TABLE 2.5 Any persistent storage system of data (i.e. a database) has basic CRUD functions.

	Operation	SQL function
C	Create	CREATE and INSERT
R	Read	SELECT
U	Update	EDIT
D	Delete	DELETE

SQL is designed to communicate with a database using programmed commands called **queries**.

To create a new database and table, use keywords.

```
CREATE DATABASE db_name
CREATE TABLE table_name (column_1,column_2, ..)
```

Add new records (a new row).

```
INSERT INTO table_name
VALUES (value1,value2,value3, ..);
CREATE DATABASE staff_db;
CREATE TABLE employees (id INTEGER, first_name VARCHAR(10), last_name
VARCHAR(10), department VARCHAR(10), salary FLOAT(4));
INSERT INTO employees
VALUES (1,'Napier','Green','ART',1675), (2,'Ash','Gray','SALES',1845);
```

Each entry is a new row and is given a unique identifier (id).

The simplest query is to select all using an asterisk (shift-8). This query will get all columns from the employees table.

```
SELECT *
FROM employees
```



Pronunciation: many (English-speaking) professionals refer to 'seek-well', while the official standards declare that the database be known as 'ess-que-ell'.

Id	first_name	last_name	department	salary
1	Napier	Green	ART	1675
2	Ash	Gray	SALES	1845
3	Cadmium	Orange	MARKETING	1785
4	Floral	White	ART	1765
5	Otter	Brown	MARKETING	1955
6	Smokey	Topaz	MARKETING	1875
7	Rose	Quartz	ART	1725
8	Caput	Mortuum	RESEARCH	1925
9	Han	Purple	SALES	2125
10	Unmellow	Yellow	SALES	1985

Queries

The syntax of the query requires the columns to be specified and the name of the table and any conditions or filters. The output can be arranged into alphabetical order (either ASC for ascending or DESC for descending):

```
SELECT {column_1}, {column_2} FROM {table} WHERE {column_1} condition
ORDER BY {column_2}.
```

For example:

```
SELECT
    first_Name,
    last_Name,
    salary
FROM employees
WHERE salary > 1500 AND salary < 3000
ORDER BY last_Name ASC
```

Note: SQL can use conditional logical operators AND, OR and NOT to test the filtering condition to be true.

TABLE 2.6 Query output

first_name	last_name	salary
Otter	Brown	1955
Ash	Gray	1845
Napier	Green	1675
Caput	Mortuum	1925
Cadmium	Orange	1785
Han	Purple	2125
Rose	Quartz	1725
Smokey	Topaz	1875
Floral	White	1765
Unmellow	Yellow	1985

Calculations

The aggregate function SUM() and GROUP BY can find totals by category.

```
SELECT
    department,
    SUM(salary) AS total_salary
FROM employees
GROUP BY department
```

Note: Any non-aggregated column in SELECT must also be in GROUP BY.

TABLE 2.7 Query output

department	ART	MARKETING	RESEARCH	SALES
total_salary	5165	5615	1925	5955

Other basic functions available include:

- COUNT()
- MIN() – minimum value
- MAX() – maximum value
- AVG() – average value, which is the result of SUM() divided by COUNT().

When combined with GROUP BY, these functions produce query output in categories.

For example, find the number of employees in each department and then the average salary for each department.

```
SELECT
    department,
    COUNT(*) AS number_of_employees
FROM employees
GROUP BY department
```

TABLE 2.8 Query output

department	ART	MARKETING	RESEARCH	SALES
number_of_employees	3	3	1	3

```
SELECT
    department,
    AVG(salary) AS average_salary
FROM employees
GROUP BY department
```

TABLE 2.9 Query output

department	ART	MARKETING	RESEARCH	SALES
average_salary	1722	1872	1925	1985

SQL wildcard symbols substitute for one or more string characters.

There are two defined wildcard characters used in LIKE and NOT LIKE operations:

- `_` underscore represents a single character
- `%` represents zero or more characters.

The LIKE operator searches for a specific pattern in a column.

```
SELECT * FROM employees
WHERE first_name LIKE 'A%'
```

TABLE 2.10 Query output

Id	first_name	last_name	department	salary
2	Ash	Gray	SALES	1845

Query designs

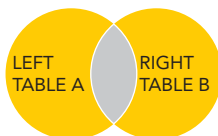
JOINS

A relational database management system (RDBMS) is often comprised of many tables. SQL has a means of combining tables, referred to as JOINS. There are four main combinations, though a FULL JOIN is rarely used and is included here for completeness (Figure 2.14).

JOIN clause allows the combination of rows from two or more tables based on a related column

(INNER) JOIN – intersection

a combined file (A + B) where a common field exists, i.e. NULL fields are removed



LEFT (OUTER) JOIN

takes all records from table A and any matches from table B



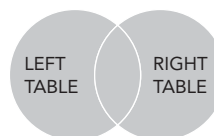
RIGHT (OUTER) JOIN

takes all records from table B and any matches from table A



FULL (OUTER) JOIN

takes all records from table A and all records from table B, whether NULL or not



There are detailed descriptions and further examples of SQL available on Nelson MindTap.

There are other types of JOIN:

- cross JOIN
 - JOIN with wildcard
 - JOIN on foreign key.
- If you intend using SQL in Applied Computing Units 3 and 4, then exploring how these operate would be a good idea.

FIGURE 2.14 SQL combines tables using JOIN.

Here is an example of SQL JOINS if you have two tables, TABLE_1 and TABLE_2:

TABLE_1

item_id	column_1
1	A
2	B
4	D
6	F
7	G

TABLE_2

item_id	column_2
1	A
3	C
5	E
6	F
8	H
9	I

(INNER) JOIN
table_1 and table_2

1	A
6	F

LEFT (OUTER) JOIN
table_1 and table_2

1	A
2	B
4	D
6	F
7	G

RIGHT (OUTER) JOIN
table_1 and table_2

1	A
3	C
5	E
6	F
8	H
9	I

FULL (OUTER) JOIN
table_1 and table_2

1	A
2	B
3	C
4	D
5	E
6	F
7	G
8	H
9	I

Using artificial intelligence to generate SQL queries

Artificial intelligence, such as ChatGPT by OpenAI, can write SQL queries. There are clear advantages and a few disadvantages with this approach, as seen in Table 2.11.

TABLE 2.11 Advantages and disadvantages of using GPT to generate SQL queries

Advantages	Disadvantages
ChatGPT generates SQL queries quickly and, when used by experienced SQL analysts, will save time.	ChatGPT requires specific prompts for useful output.
Beginners already know how to use a chatbot.	ChatGPT has no context knowledge.
ChatGPT already knows SQL rules and syntax.	Security liabilities are created when db details are input as a prompt, as they can be used in other output responses.
	ChatGPT has no intuition for enquiring into SQL.
	ChatGPT is imprecise with actual output. The query prompt may be technically correct, but not achieve the intended purpose.

Steps for prompting ChatGPT to generate SQL output:

- 1 Devise a simple SQL problem without any unnecessary extra details.
- 2 Explain the schema details, structure of tables, names of columns, links between tables, which version of SQL is being used.
- 3 Specify any limitations and describe how the expected output will be used and by whom.
- 4 Run the prompt query, inspect the output, modify the prompt and run the prompt query again as often as necessary until the output is acceptable. This is an iterative process, and is expected to narrow into the desired result after several attempts.

Consider the following example of prompts being given to ChatGPT to write a SELECT statement.

Input prompt:

I need an SQL query for ABSdata2021 that lifts from Table Total_Weekly_Income and returns COUNT which meets the filter less than \$95k and lives in Victoria with postcode in the three thousands.

You
I need an SQL query for ABSdata2021 that lifts from Table Total_Weekly_Income and returns COUNT which meets the filter less than \$95k and lives in Victoria with postcode in the three thousands

ChatGPT
Assuming you have a table named "Total_Weekly_Income" in the "ABSdata2021" database with columns like "income," "state," and "postcode," you can use the following SQL query:

```
sql
SELECT COUNT(*)
FROM Total_Weekly_Income
WHERE income < 95000
AND state = 'Victoria'
AND postcode BETWEEN 3000 AND 3999;
```

This query selects the count of records from the "Total_Weekly_Income" table where the income is less than \$95,000, the state is "Victoria," and the postcode is in the range of 3000 to 3999. Adjust the column names accordingly based on the actual structure of your dataset.

Selina Dennis

FIGURE 2.15 ChatGPT SQL output

Extracting data

Data that is relevant to specific purposes can be obtained from data repositories such as databases and spreadsheets.

Spreadsheet software

Many data sources provide data in a file format that is compatible with existing spreadsheet software applications.

Figure 2.16 shows a data set available from the AFL Victoria website, containing all of the details and locations of AFL teams and programs in Victoria. The data set is available in CSV file format that is compatible with spreadsheet software.



Weblink
AFL Victoria

FIGURE 2.16 Data set downloaded into spreadsheet software

Source: The State of Victoria, Department of Environment, Land, Water & Planning 2015, Creative Commons Attribution 3.0 Australia

	A	B	C	D	E	F	G	H	I
1	Name	Address	Suburb	Postcode	State	Category	LGA	Region	
2	AFL Auskick - Aberfeldie	Aberfeldie Primary	Essendon	3040	VIC	AFL Auskick Moonee Valley		North and West Metropolitan Region	
3	AFL Auskick - Airport West	Hansen Reserve	Airport West	3042	VIC	AFL Auskick Moonee Valley		North and West Metropolitan Region	
4	AFL Auskick - Ajax	Princes Park	Caulfield South	3162	VIC	AFL Auskick Glen Eira		Southern Metro	
5	AFL Auskick - Alexandra	5665 Maroondah	Alexandra	3714	VIC	AFL Auskick Murrindindi		Hume	
6	AFL Auskick - Alfredton	Alfredton Reserve	Alfredton	3350	VIC	AFL Auskick Ballarat		Grampians	
7	AFL Auskick - Allansford	Allansford Recre	Allansford	3277	VIC	AFL Auskick Program			
8	AFL Auskick - Alphington/Fairfield		Northcote	3070	VIC	AFL Auskick Yarra ; Darebin		North and West Metropolitan Region	
9	AFL Auskick - Altona	Altona Football C	Altona	3018	VIC	AFL Auskick Hobsons Bay		North and West Metropolitan Region	
10	AFL Auskick - Altona Meac	Comden Reserve	Altona Meadows	3028	VIC	AFL Auskick Wyndham ; Hob		North and West Metropolitan Region	
11	AFL Auskick - Amateurs	Queens Park Oval	Newtown	3220	VIC	AFL Auskick Greater Geelong		Barwon S/W	
12	AFL Auskick - Anakie	2155 Ballan Roac	Anakie	3221	VIC	AFL Auskick Surf Coast ; Gre		Grampians, Barwon S/W	
13	AFL Auskick - Anglesea	Ellimatta Reserve	Anglesea	3230	VIC	AFL Auskick Surf Coast		Barwon S/W	
14	AFL Auskick - Apollo Bay	Apollo Bay Recre	Apollo Bay	3233	VIC	AFL Auskick Colac-otway		Barwon S/W	
15	AFL Auskick - Apollo Parkways		Greensborough	3088	VIC	AFL Auskick Nillumbik ; Bany		North and West Metropolitan Region	
16	AFL Auskick - Ararat	Richardson Oval	Ararat	3377	VIC	AFL Auskick Ararat Rural		Grampians	
17	AFL Auskick - Ararat North	Blake St	Ararat	3377	VIC	AFL Auskick Program			

Spreadsheet software also contains a feature known as a web query, which allows data to be acquired from online data sources. One limitation of a web query is that the spreadsheet software needs to recognise the data.

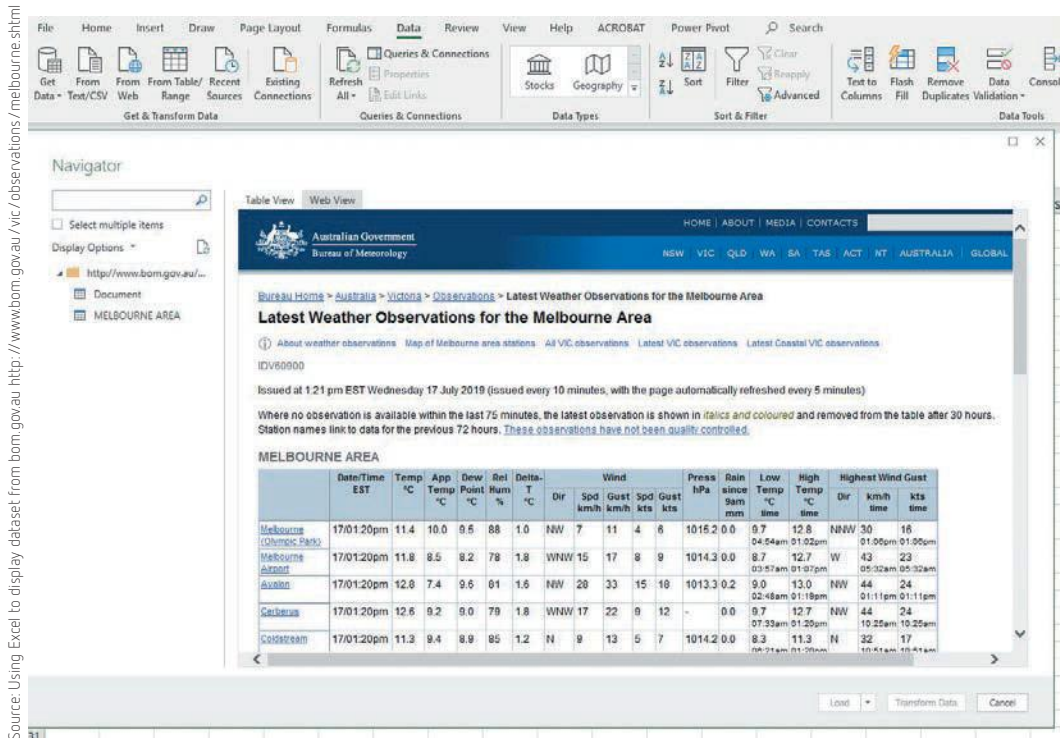


FIGURE 2.17 Example of a web query in spreadsheet software

The screenshot shows the Excel interface with a web query imported from the Australian Bureau of Meteorology. The query is titled "Latest Weather Observations for the Melbourne Area" and displays a table of weather data for various Melbourne locations. The table includes columns for Date/Time, Temp, App Temp, Dew Point, Rel Hum, Delta T, Wind Dir, Wind Spd, Wind Gust, and Wind Gust kts. The data is for 17/01/2019 at 12:00pm.

Station	Date/Time	Temp	App Temp	Dew Point	Rel Hum	Delta T	Wind Dir	Wind Spd	Wind Gust	Wind Gust kts
Melbourne (Olympic Park)	17/01/2019 12:00pm	11.4	10.2	9.7	89	0.9	NW	6	13	3
Melbourne Airport	17/01/2019 12:00pm	11.4	8.7	9.0	85	1.2	WSW	13	15	7
Avalon	17/01/2019 12:00pm	13.1	6.3	9.4	78	1.9	WNW	35	50	19
Cerberus	17/01/2019 12:00pm	12.3	8.7	8.4	77	1.9	NW	17	32	9
Coldstream	17/01/2019 12:00pm	11.2	8.8	8.6	84	1.3	N	11	20	6
Cranbourne	16/09/2019 12:00pm	10.4	7.3	8.8	90	0.8	W	15	-	8
Essendon Airport	17/01/2019 12:00pm	11.1	7.4	10.2	94	0.5	WSW	20	24	11
Fawkner Beacon	17/01/2019 12:00pm	-	-	-	-	-	WNW	33	46	18
Ferry Creek	17/01/2019 12:00pm	7.0	5.0	7.0	100	0.0	NW	7	15	4
Frankston	17/01/2019 12:00pm	-	-	-	-	-	NW	28	33	15
Geelong Racecourse	17/01/2019 12:00pm	13.2	8.1	9.3	77	2.0	WNW	26	39	14
Laverton	17/01/2019 12:00pm	11.5	7.0	9.9	90	0.8	WNW	24	33	13
Moorabbin Airport	17/01/2019 12:00pm	11.9	7.5	7.6	75	2.1	W	20	33	11
Rhyll	17/01/2019 12:00pm	11.8	8.8	7.9	77	1.9	NW	13	28	7
Scoresby	17/01/2019 12:00pm	10.4	8.4	8.5	88	0.9	NNW	9	17	5
Sheoaks	17/01/2019 12:00pm	11.1	5.5	7.2	77	1.9	WNW	26	43	14

FIGURE 2.18 Data imported into spreadsheet software using a web query

Database software

Database software can also be used to acquire data from data sources. Using the external data function, data can be imported from a range of sources and file formats, including SQL data sets and repositories, HTML documents, Excel, text and XML files, as well as external databases.

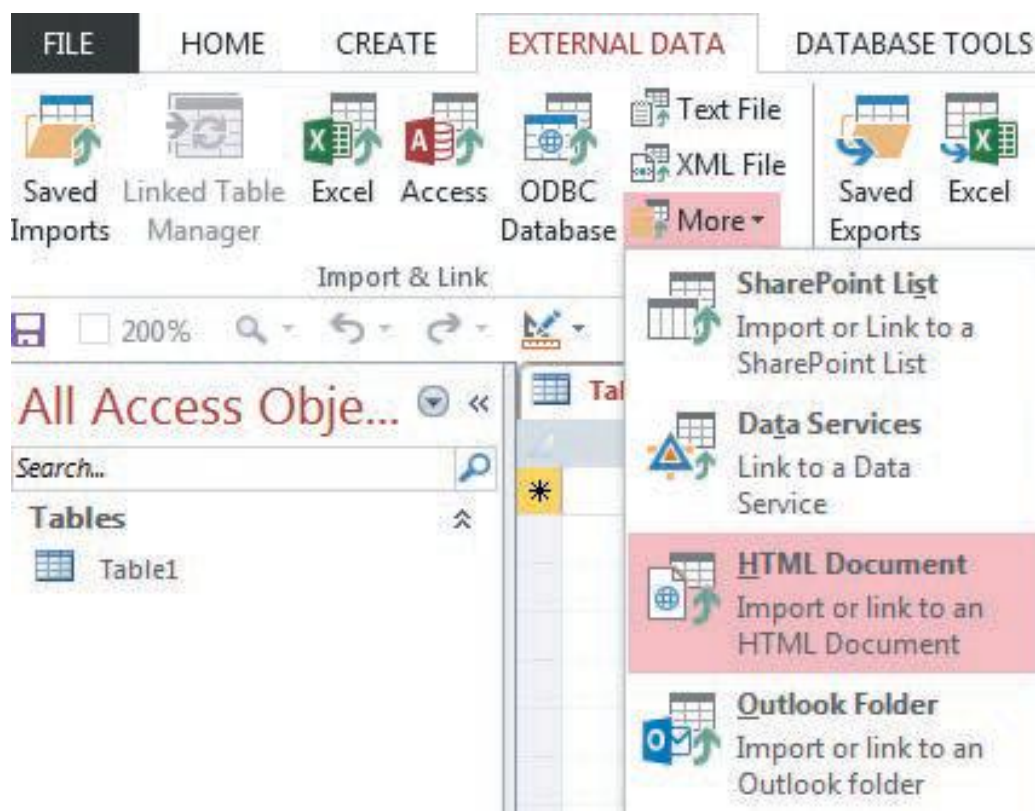


FIGURE 2.19 Options to import external data into database software

Programming languages

Some programming languages have the capability to acquire data from external sources.

Python

Python is a free programming language. It comes installed as part of macOS and must be downloaded and installed on WIN OS. There are many free libraries available that provide the ability to graph and chart data.

TABLE 2.12 Graphics plotting tool libraries for Python

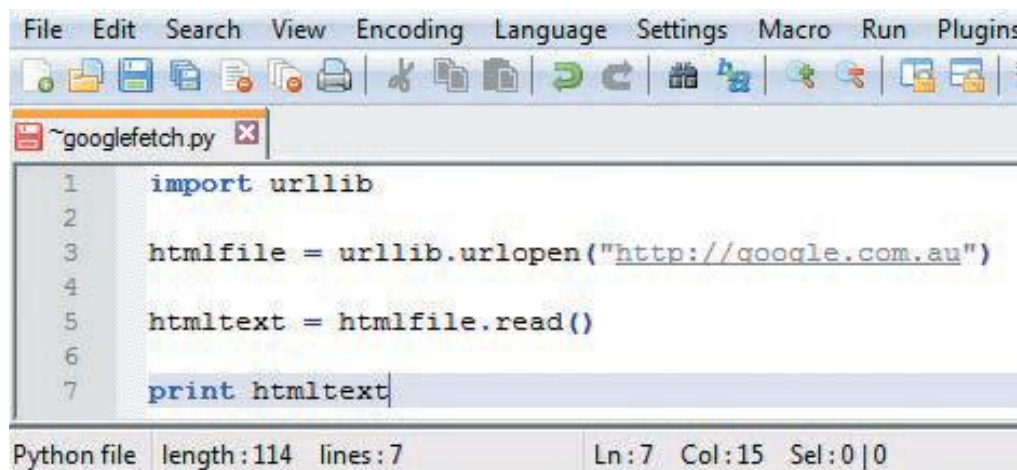
Matplotlib	This was the first data visualisation library for Python. Many other libraries make use of these functions during analysis – for example, pandas and Seaborn
Seaborn	A modern display style with improved colour palettes and formats that uses the old style matplotlib
ggplot	Copied across from ggplot2 for the R plotting system and based on concepts from <i>The Grammar of Graphics</i> . ggplot works closely with pandas
Bokeh	A Python native library that does not rely on other libraries, also based on <i>The Grammar of Graphics</i> . There are no default settings, so every element is required to be defined
Pygal	Create interactive plots that can be embedded in HTML to be displayed by a web browser. Suitable for small data sets
Plotly	Python can access the online plot.ly to create interactive graphics and offers a greater range of chart options
Geoplotlib	A dedicated library that creates maps

Source: wiki.python.org

Python can fetch (extract) data from websites. Applications can be created in Python that will fetch data from a website based on the page's HTML code.

Using the 'Inspect Element' command in a browser, you can view the HTML code of a web page. The area that the data is located in the code can be identified and then imported into another application. Once data has been acquired, it can be imported into other applications, such as database or spreadsheet software.

Figure 2.20 shows a sample of Python code that will read all of the HTML code from Google.



```

1  import urllib
2
3  htmlfile = urllib.urlopen("http://google.com.au")
4
5  htmltext = htmlfile.read()
6
7  print htmltext

```

Python file length:114 lines:7 Ln:7 Col:15 Sel:0|0

FIGURE 2.20 Sample Python code reading HTML code from Google

Mathematica

Mathematica uses Wolfram Language, which has inbuilt functions to display data acquired in real time from online 'databins' using the data drop website. Mathematica uses online processors so the local machine does not do the computation. Every possible graphing style and function is built in with default settings. Detailed documentation with examples is provided on every feature.

What is the difference between Python and Mathematica (and R)?

Python is free and open-source. Mathematica is a proprietary application and has an annual cost associated with using the local installed program and the online Wolfram cloud. Matplotlib is a competing proprietary application that is used in engineering studies, while Mathematica is used in mathematics and physics studies. R programming language is also free and open-source. R requires a deep programming knowledge and is considered too advanced for introductory secondary school purposes. Whether you choose a detailed programming approach with Python or a 'black box' approach with Mathematica will depend on a number of local factors, including:

- familiarity with coding Python or Wolfram Language
- time available to devote to understanding a new language
- intended further study. For example, will the knowledge be useful in later studies?
- teacher preference. Students are often guided by the amount of support that their school and teacher can provide.

Validation of data

To make the solution accurate and reasonable, all data that is used should be validated. This means that the original data should be manually checked for illegal data types, for reasonableness, for correct spelling, to ensure that data falls within a correct 'range' or that any codes that are used are consistent (follow similar style) and reasonable (similar codes relate to similar products).

Wolfram Data Drop collects data in real time from a variety of user-defined sources.



Weblink
Wolfram Data Drop

Mathematica and Wolfram Language are available to all Victorian students and teachers at no cost. Further details are available online. Web search 'Mathematica free Victoria Wolfram software'.

THINK ABOUT APPLIED COMPUTING

2.2

Steps

- 1 Open a new Excel file.
- 2 Choose a small range of cells.
- 3 Apply data validation rules to those cells for date, time, number, etc.
- 4 Test that your data entry conditions work as expected.

Questions

- 1 What happens when you apply validation rules to existing data already entered into cells?
- 2 How may this assist your data manipulation and preparation for data visualisation charts?

The spreadsheet standards review board provides best practices for spreadsheet modelling. The Best Practice Modeling Guidelines (BPMG) are also used.



Weblinks
Spreadsheet
Standards Review
Board (SSRB)

Modano

Web search 'Excel data forms'. The most reliable advice will begin with URL 'support.office.com'. However, there are many other tutorials and comments that you may find useful.

It is easy for data entry errors to occur during the input phase, particularly if a large amount of data is involved. Data should therefore always be validated before any processing occurs.

There are two types of **validation**: manual and electronic validation.

Manual validation occurs when the data entered is checked for accuracy by a person rather than by a machine. Proofreading is one manual validation technique. When you proofread data, you look for transcription errors. The data entered should be compared with the source document to ensure that they match. Any differences observed must be followed up.

Data may be validated by computer if a validation function is contained in the software or built into the solution to a problem. **Electronic validation** techniques, sometimes called machine-validation techniques, ensure the accuracy of data and are built into software, such as spell checkers. Types of electronic validation include range checking, existence checking and data type checking. You will cover these in more detail in Chapter 4.

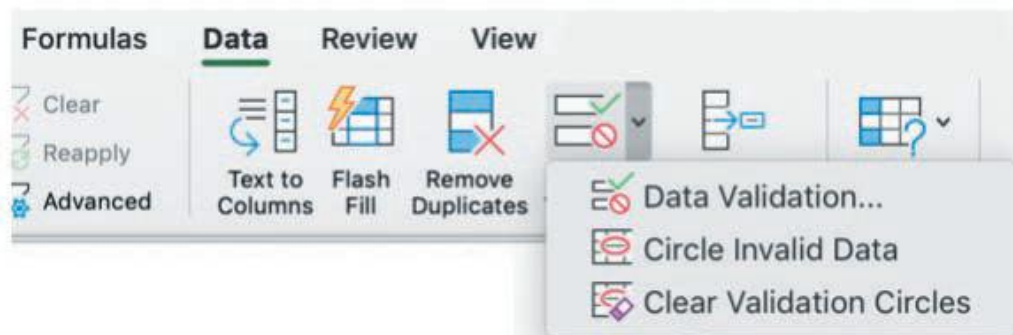
A feature of modern software is the inbuilt functions that provide assurance that the data entered will be within predetermined ranges. This validation is simple to incorporate into your design. Usually a warning to the data entry operator will ensure that the data is rejected, reviewed and correctly entered. This will create a high level of confidence in the quality of the data.

Validation applies to both databases and spreadsheets when entering data. Data visualisation software is designed to use data from other sources, so validation may already have occurred. **Filtering** of existing data is more likely in that case.

Spreadsheets

In Excel, forms can be established when there is repetitive data entry. The cells can be formatted to ensure that the data conforms to expected ranges. Certain values are disallowed.

Alternatively, a range of cells can be identified for a particular function. (See margin note and weblinks to gain access to further details.)



Selina Dennis

FIGURE 2.21 In Excel, choose Data > Data Validation. You are then prompted to insert a restriction, which will be applied to any attempt to place data in that cell.

Excel for Windows has additional functionality that includes ActiveX and data entry forms. It should be noted that a spreadsheet is *not* a database. A table of records, where each row is a record and each column is a variable or category, may be sufficient for many purposes when there are fewer than a million records. As well, Visual Basic is a scripting language that allows developers to further enhance spreadsheets.

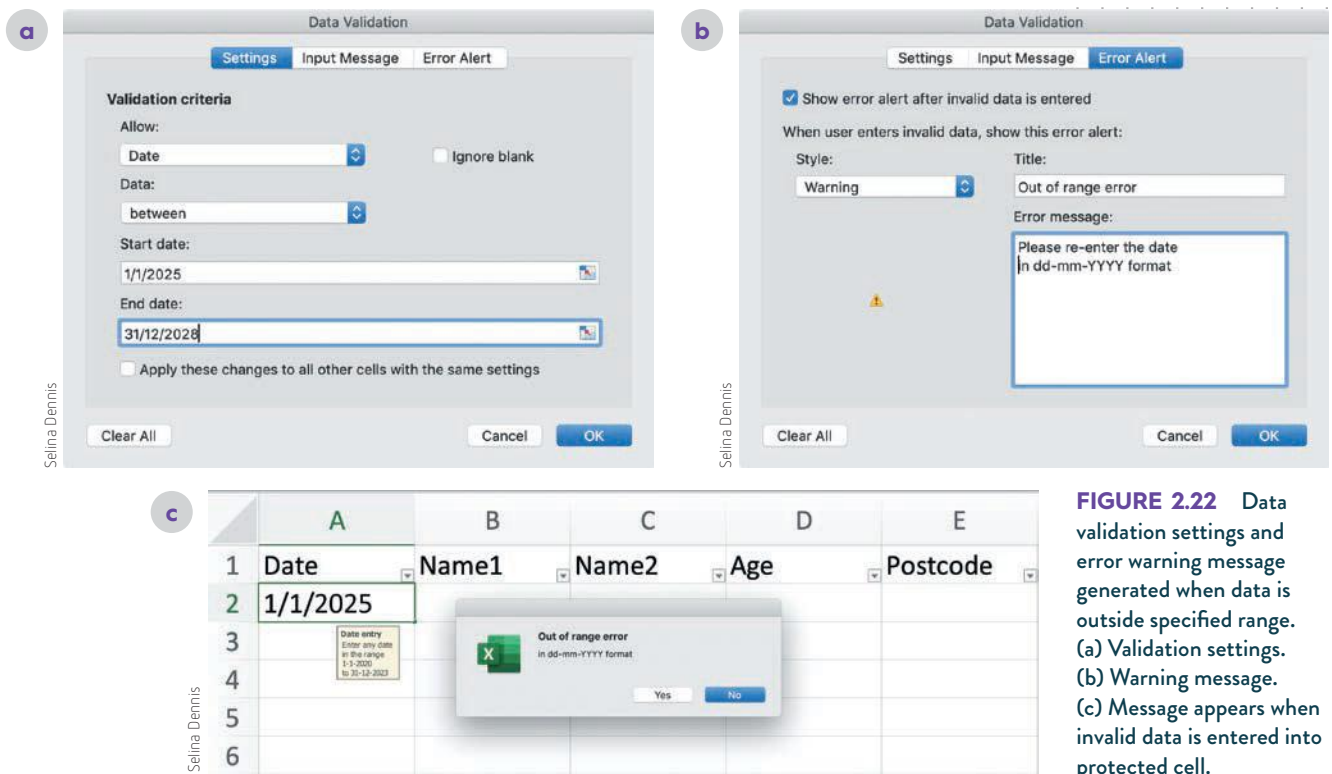


FIGURE 2.22 Data validation settings and error warning message generated when data is outside specified range. (a) Validation settings. (b) Warning message. (c) Message appears when invalid data is entered into protected cell.

Databases

There are several database software applications relevant to VCE Applied Computing, with others available at a significant cost.

Free database applications include:

- LibreOffice BASE
- MySQL (part of WAMPP/XAMPP installation, which also includes PHP and WordPress servers)
- SQLite
- Python-mysql (with prior experience)
- Python-R (very advanced)

Database applications for purchase include:

- Microsoft Access and Microsoft SQL
- FileMaker
- Wolfram Language – Mathematica.

The following examples will be in FileMaker, but the ideas apply equally to any database.

The screenshot shows a FileMaker database entry screen for a task. The title is 'AC-Networking task'. The 'Date' field is set to '26 May 2020', 'Status' is 'In Progress', and 'Category' is 'U1 - Networks'. The 'Description' field contains the text 'Assemble wi-fi extender and test with signal strength test on iPhone CODE:'. There are two sections for 'Assignees' and 'Attachments'. The 'Assignees' section has columns for 'Name', 'Email', and 'Phone'. The 'Attachments' section has columns for 'File' and 'Comments'. There are buttons for 'Add Assignee' and 'Add Attachment'. The top of the screen shows a toolbar with 'Records', 'Find', 'Sort', and 'Share' buttons. The bottom of the screen shows a 'Browse' button.

FIGURE 2.23 Database entry screen: validation for date, drop-down menu for category, open field for comment

Once the data has been successfully entered, reports can be requested from the data set. The simplest report is to list all records and all fields. The strength of the database is the ability to filter on one field and assemble a report using a selection of the other fields.

The screenshot shows a FileMaker database list view for a task. The title is 'Tasks'. The table has columns for 'Task Name', 'Category', 'Status', and 'Date'. The 'Find' button in the toolbar is highlighted with a red circle. The table shows two records: 'AC-Networking task' and 'AC-DataVis'.

Task Name	Category	Status	Date
AC-Networking task	U1 - Networks	In Progress	26 May 2020
AC-DataVis	U1 - DataVis	Completed	15 Apr 2020

FIGURE 2.24 List view indicates fields that could be used by a FIND filter.

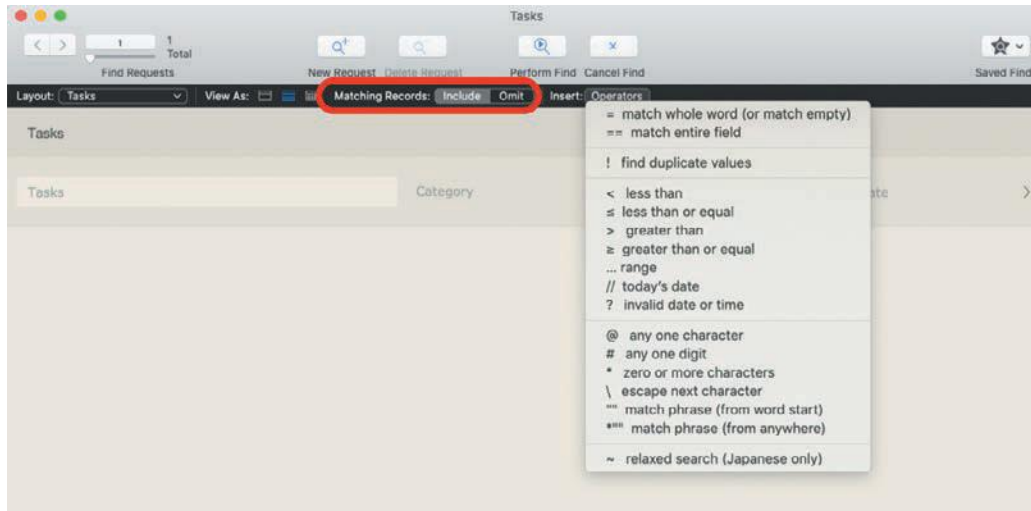
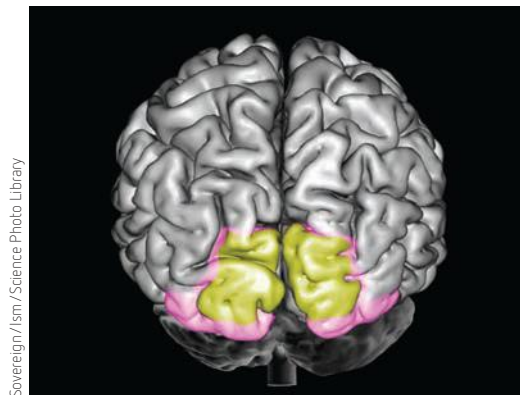


FIGURE 2.25 The FileMaker FIND filter chooses whether to include or omit the matches in the view/report.

Data visualisations

Data visualisation results from using software functions and techniques to access and select data from large data repositories, and then present that data in graphical form. This can be in the form of graphs and histograms, charts and maps, or spatial relationship diagrams and network diagrams. This type of graphic representation helps the viewer to identify patterns and relationships in vast amounts of data that would not be identifiable in the raw data. Data visualisations can be still (**static**) or moving, or updated regularly or in real time (dynamic). Emerging examples using virtual reality (VR) and augmented reality (AR) allow viewers to be immersed within the data to observe patterns and relationships. Chatbots with AI offer the promise of greater capacity to interpret data with detailed explanations and graphics.

While there are many ways to convey information based on data, a visual method is the most effective. Extensive research has consistently supported the proposition that humans gain better understanding by visual communication. The human brain has developed to interpret visual data quickly and accurately.



Sovereign/ism/Science Photo Library

When should you choose a spreadsheet and when a database?

The term 'big data' initially referred to data sets that could not fit into the RAM on a local computer. In our case, Excel is restricted by available RAM to 1048576 (rows) × 16384 (columns = XFD) or about 10 GB file size.

With the development of online data streaming, 'big data' can also be described by the three 'V factors':

- **volume** – large amounts of unstructured data that could have a file size of tens of terabytes or hundreds of petabytes
- **velocity** – the rate at which the data is received. Some applications are processing data in near real time
- **variety** – unstructured data consisting of a variety of data types, including text, images, audio and video, which require processing to derive meaning from the metadata.

FIGURE 2.26 The human brain has a preference for visual information. The primary visual cortex (yellow) and the secondary visual cortex (pink) are highlighted here. The visual cortex receives and processes visual input from the retina.

There are two main aspects of visualisation communications:

- *Information visualisation* deals more with ideas and concepts.
- *Data visualisation* offers a graphic representation of complex **data sets**.

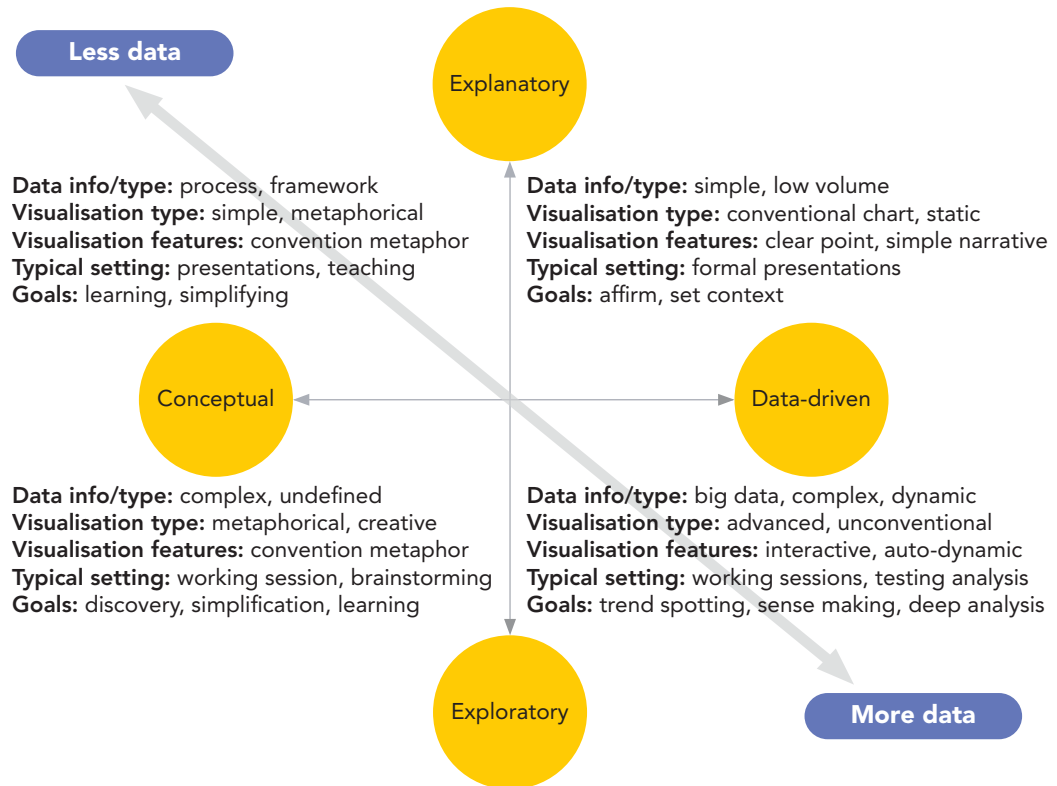
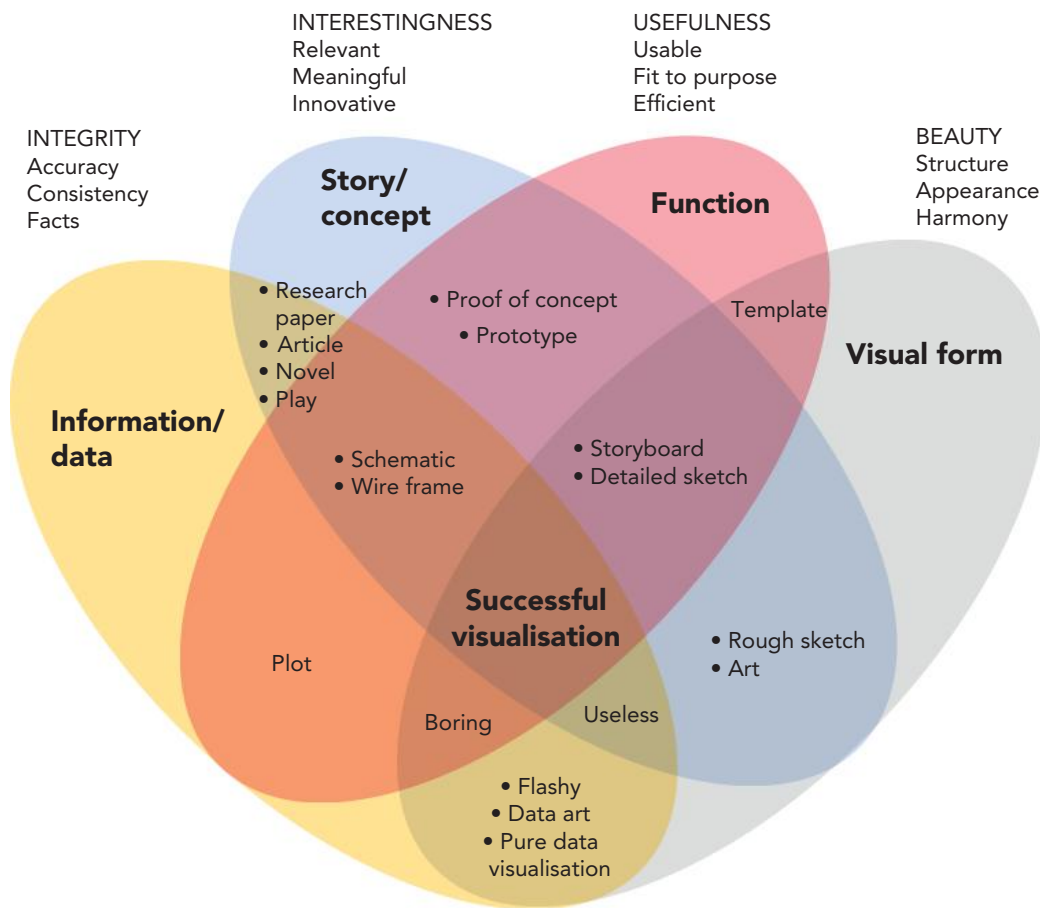


FIGURE 2.27 Data and information may be visualised in several ways.

Purposes of data visualisations

There are many purposes for a data visualisation. A successful chart could satisfy many needs by simultaneously:

- educating to develop understanding
- entertaining to amuse and distract or provoke a response
- informing to convey information
- persuading to encourage adoption of a point of view or opinion.



Source: <https://informationisbeautiful.net/visualizations/what-makes-a-good-data-visualization>

FIGURE 2.28 What makes a good visualisation?

Categories of data visualisations

Data visualisations can be loosely described in two categories:

- *explanation* – to convey an existing point of view. This style may also be persuasive, inviting the viewer to reconsider a previously held point of view
- *exploration* – allows for some user choice. The views are selected from a range of options. The combinations of views, while necessarily limited, are not predetermined. The viewer may locate and further interrogate a data item of interest.

Explanatory

An explanatory data visualisation sets out to explain a point of view. Features of this type of visualisation are that it is:

- clearly expressed
- simply laid out
- based on known facts.

Spelling varies according to whether AU/UK or US English is being used. In this text, we apply the AU/UK spelling 'visualisation', rather than the US spelling 'visualization'. Curiously, when abbreviated, the AU/UK 'dataVis' is consistent with both spellings, while the alternative 'dataViz' is not.

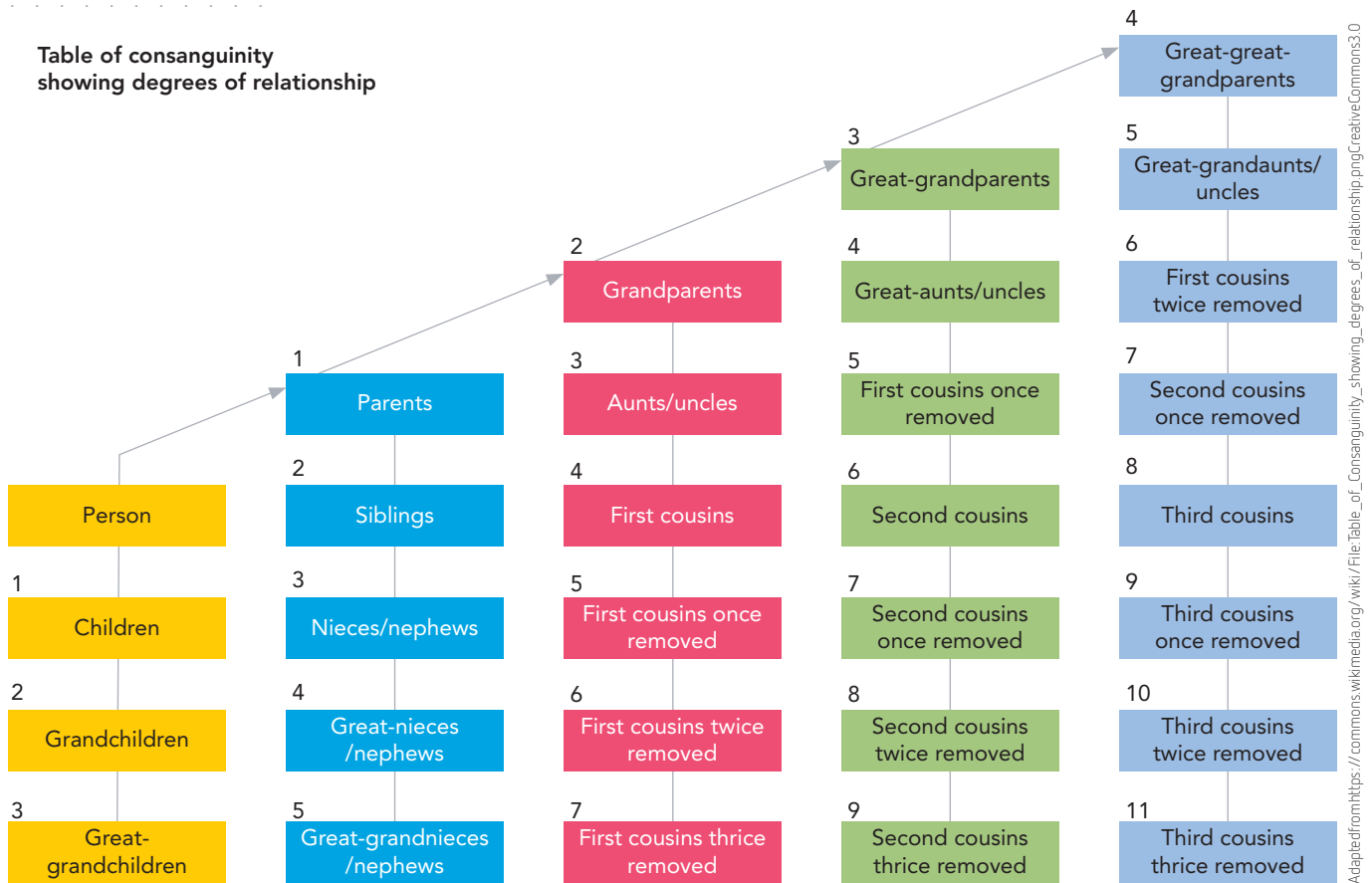


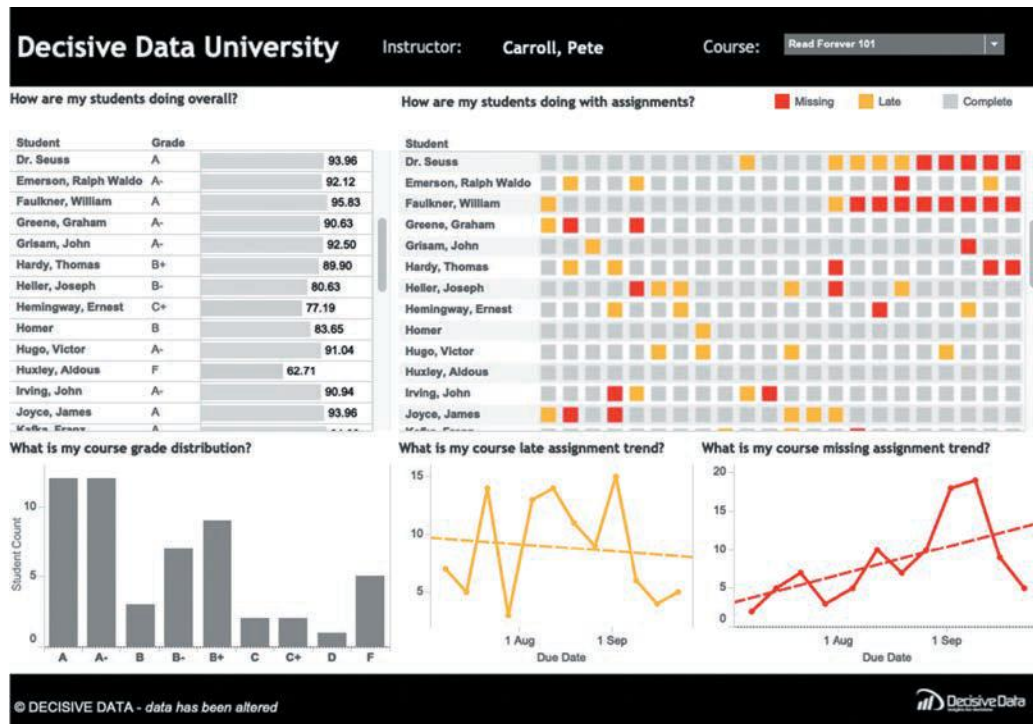
FIGURE 2.29 There is very little number data here. Consanguinity shows the degree of separation from a person of interest across a family's generations, known as 'how many times removed'.

Exploratory

Exploratory visualisations allow users to select a value or category to explore connections and relationships. An example is shown in Figure 2.30, which is an on-screen interactive visualisation of student progress and results. The relationships between submitted work, progress and grades for the entire class group can be combined onto one screen. The hover tooltip reveals further details on each record. As further records are added, the display is recalculated and updated.

Informing by demonstration or exhibition

Demonstration or exhibition visualisations allow for demonstration of relative data values and comparison between categories. On-screen, these may be interactive to allow for user input and direction.



Source: <https://www.decisivedata.net/k12-classroom-evaluation>

FIGURE 2.30 An interactive dashboard with data visualisation. This Tableau demonstration has hover tooltips for each element to provide further detailed information.

Types of data visualisations

Data visualisations come in various forms, each suited to different purposes and audiences. In VCE Applied Computing, you will consider infographics (long-form and series) and dashboards.

Infographics

Infographics are designed to present complex information in a visually appealing and easily digestible format. Long-form infographics are detailed visual representations that combine text, images and various data visualisation elements like charts and graphs to convey a comprehensive narrative on a specific topic. They are often used to educate and inform audiences by breaking down complex information into easily digestible segments, making the data engaging and accessible.

A series of posters of infographics presents related information in a segmented and focused manner, usually displayed sequentially. Each poster in the series covers a specific aspect of a broader topic, allowing for in-depth exploration of each segment. This method is particularly effective for educational purposes or public awareness campaigns, as it breaks down complex data into more manageable and comprehensible parts.

Characteristics of an infographic poster series

- All posters within the series are related to a central theme or topic. For example, a series on climate change might include posters on causes, effects and solutions.
- Each poster addresses a different facet of the overall theme, allowing for detailed exploration of specific subtopics. This helps to break down complex information into more manageable parts.

View an example of a long-form infographic on the evolution of computer science via the Visual Capitalist website.



Weblink
Visual Capitalist

Online infographics can be beautiful. Consider some of the different methods of visualising data that are displayed on the Information is Beautiful website.



WebLink
Information is Beautiful

- The design style, colour scheme and layout are consistent across all posters in the series, creating a unified look and feel. This consistency helps viewers recognise and associate the posters as part of a coherent set.
- Posters often use a combination of visual elements such as charts, graphs, illustrations and minimal text to make the information easily digestible.
- Posters can be displayed sequentially in a physical space, such as a hallway, classroom or exhibition or they can be viewed individually in digital formats.

Dashboards

Dashboards are interactive data visualisation tools that provide real-time insights into key metrics and performance indicators. They typically display multiple types of visualisations with varied components on a single screen, allowing users to monitor and analyse data efficiently. Dashboards are widely used in business and data analytics for tracking progress, identifying trends and making informed decisions.



FIGURE 2.31 Data visualisation dashboard

Components of data visualisations

Any graphic that displays and explains information, whether data or words, can be described as a 'data visualisation'. Data visualisations can take many forms. The main components used to convey the message can include:

- text and graphics
- tables
- charts and graphs.

Graphics can include any of the following:

- animations
- audio
- video.

The first charts were produced in response to a need to convince others about a situation. The visual presentation was a means of showing numbers that were otherwise too difficult to follow and whose significance was therefore lost.

Florence Nightingale, for example, was not only a nurse, but a statistician and a social reformer. She made use of a **rose chart** or **coxcomb chart** – a combination of stacked bar and pie chart used to convey proportion and amount in one image (see Figure 2.32). She used this to convince hospital administrators to change hygiene practices during the British war in the Crimea (1853–56) (modern-day Ukraine).

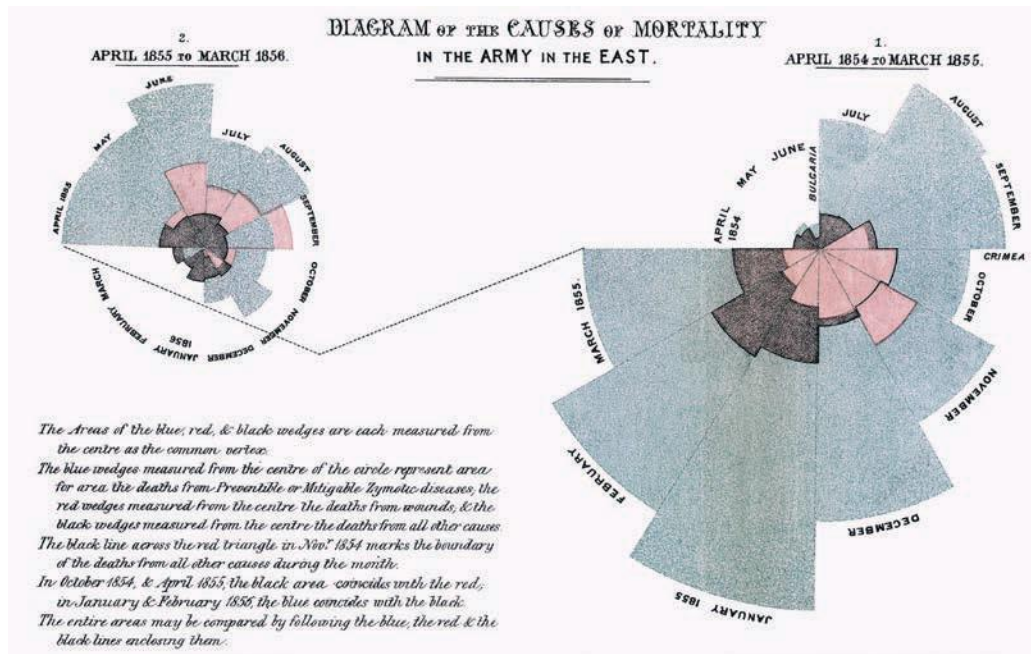


FIGURE 2.32

Florence Nightingale provided persuasive rose chart analysis arguing that disease caused more deaths than wounds in the Crimean War (1853–56).

Chart with a story

Widely regarded as the best graph ever drawn, Charles Minard's visual representation of Napoleon's march to Moscow (Figure 2.33) gives the user information on several levels at once:

- geographical
- climatic
- numerical.

The top line represents the troops moving (from left to right) towards Moscow in 1812–13. The figures on the side of the line show the dramatic fall in troop numbers, and the corresponding thinning of the line shows the fall visually. It can be seen that each time Napoleon crossed a river, he lost large numbers of troops. The line at the bottom of the graph shows temperature. As the temperature continued to drop, still more troops were lost.

The black line shows the troops retreating (from right to left). One can only speculate about how different history might have been if Napoleon had had Minard in his midst during his troop campaign in Russia.

Thirty years later, Matthew Henry Phineas Riall Sankey used this visualisation for energy flow. This type of graph came to be called a **Sankey diagram**. These band graphs are routinely used for illustration of flows.

CASE STUDY



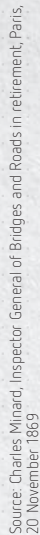


FIGURE 2.33 Charles Minard's map showing the French army's disastrous Russian campaign of 1812. The graphic represents six types of data in just two dimensions: the number of Napoleon's troops; distance; temperature; the latitude and longitude; direction of travel; and location relative to specific dates.

Charts

A **chart** (also known as a graph) is a method of displaying data visually, where the data set is represented as symbols in the chart. Many spreadsheet applications have charting (or graphing) capabilities. Features of a chart can include a title, axis, scale or grid, data labels and a legend.

Charts are often used to visualise numerical data. There are a range of chart types, and each type is used for different purposes. A bar chart can be used to compare different items, a pie chart to show each data item as a proportion of the population, line charts are useful for showing the trend in a data item over time and histograms are useful for grouping data, then showing the frequency of each group. New types of charts are regularly developed that have not been seen previously as animation and presentation technologies allow greater functionality.

Which chart to choose?

It is your choice how best to present the data you have selected to present, illustrate and support a statement of your findings. There is no best way, but there are other, better ways of presenting the data and information.

In making your decision on how to present the data, consider how you will:

- convey a message or meaning
- identify trends or specific information
- illustrate an aspect worthy of attention
- provide an overall view of the data.

Different types of charts have advantages over others for achieving clearer communication. Figure 2.34 shows different types of charts used for different purposes, but these only scratch the surface in terms of types of data visualisations.

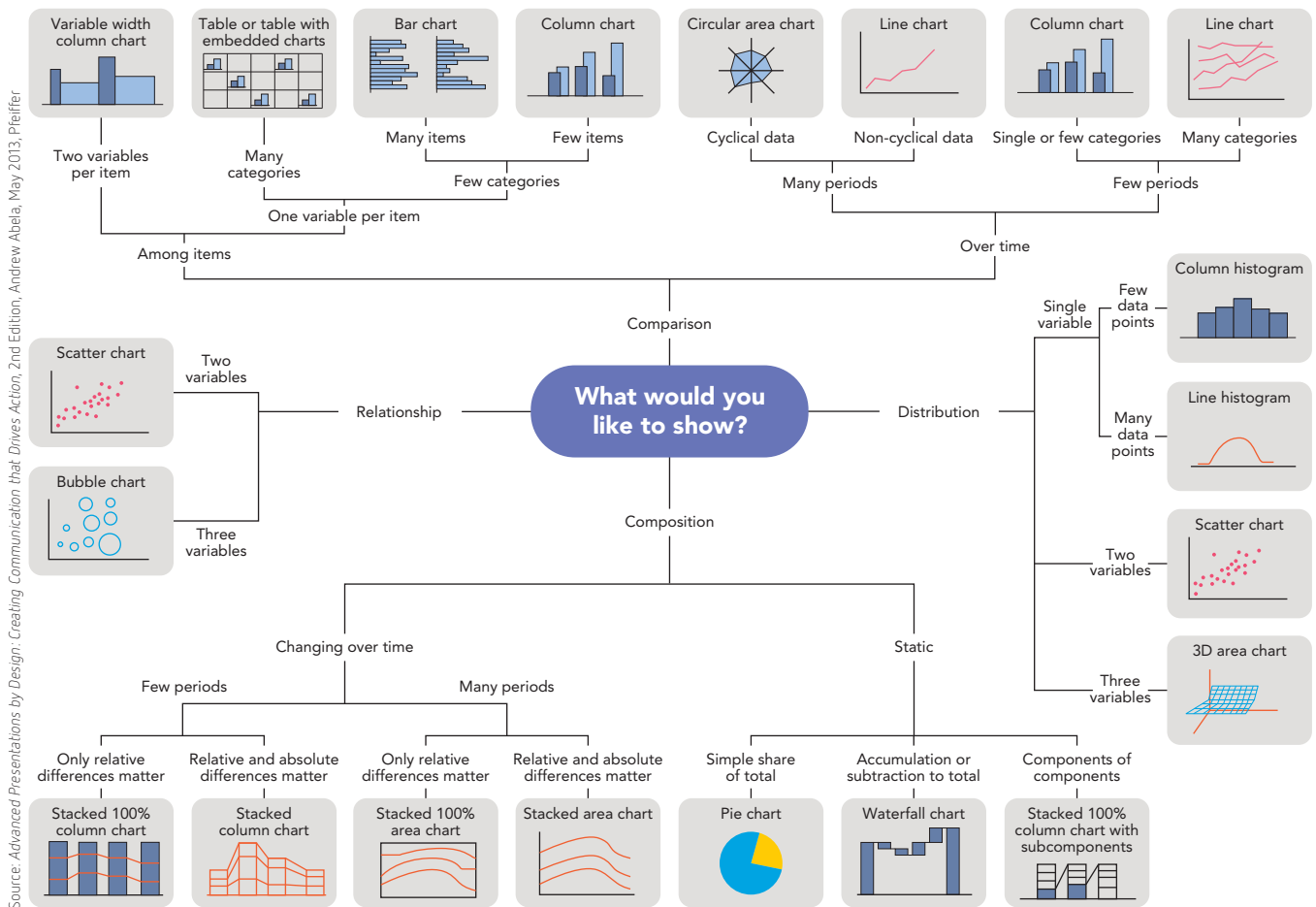


FIGURE 2.34 Some common types of data visualisations

Qualitative data presents some difficulties as charts using standard axes do not easily display such data in a meaningful way. The Qualitative Chart Chooser website has a gauge to determine which chart may be appropriate.

A quick word about truthful graphs

Any graph type should be considered if it can accurately and truthfully convey insight or a message. There are, however, several graphs that consistently mislead and confuse.

Pie charts

A **pie chart** shows numerical proportions of a whole entity. If a pie chart is to be truthful, it should only be used with fewer than six categories and be presented with a 2D aspect.

When choosing to use a pie chart, ensure that the parts add to 100 per cent, and that the pieces of pie are ordered from smallest to largest, beginning at 0° (the top of the circle), and moving clockwise (Figure 2.35).

Three-dimensional pie charts can be misleading. Three dimensions represented on two-dimensional media must distort the information contained by the graphic. These distortions may misrepresent the data and create perceptions that are wrong in fact (Figure 2.36).



Weblink
Qualitative Chart
Chooser

'If the only thing worse than a pie chart is lots of them, then I think I've discovered that the only thing worse than lots of them, is half of one.'

Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Cheshire, Connecticut: Graphics Press.

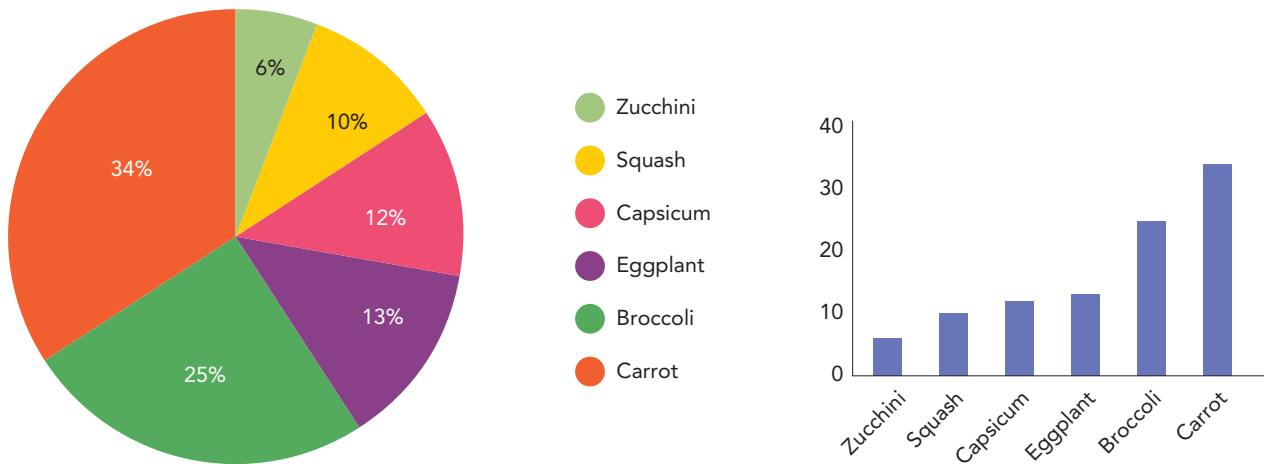


FIGURE 2.35 An example showing the limitations of bar charts and pie charts for quick accurate comparisons. A limitation of the bar chart is that it is not immediately apparent that it represents a total of 100 per cent, so each bar is a proportion of the whole.

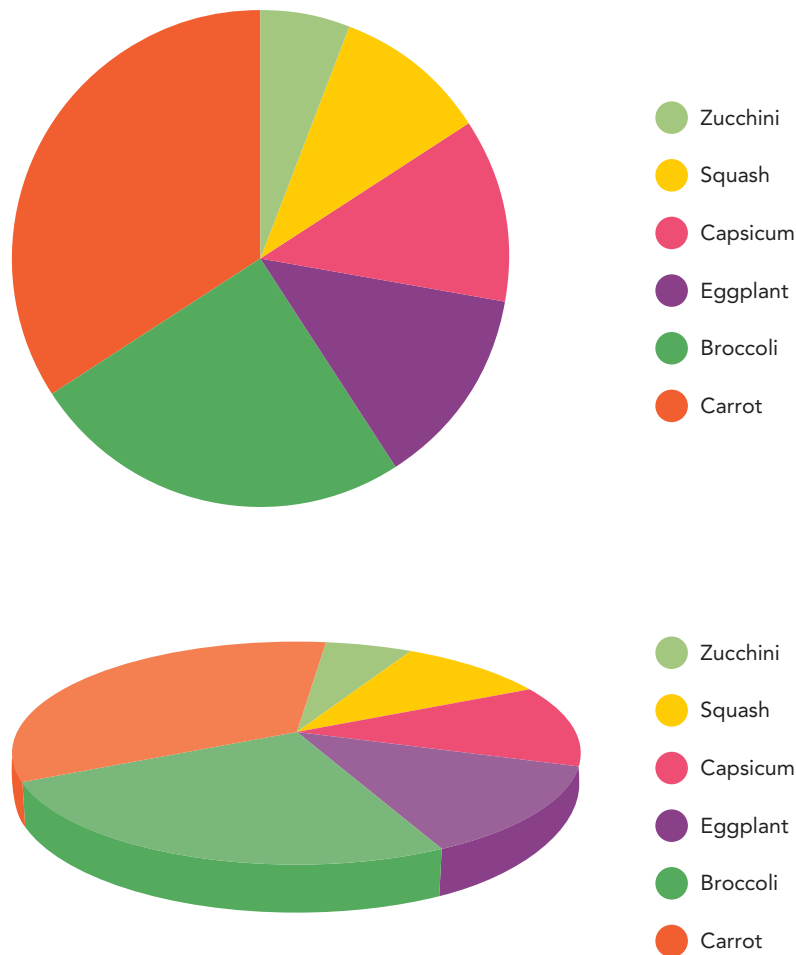


FIGURE 2.36 The 3D pie chart appears to show carrot (orange, 34 per cent) is smaller than broccoli (green, 25 per cent). The angle to the eye is larger for the sector at the lower edge of the visual. This distortion creates a false impression.

The most glaring 3D to 2D distortion is the mapping of Earth's globe in two dimensions, where areas of countries are stretched or compressed. Africa and Australia are consistently reduced in size, and Europe and North America are enlarged. This creates a wrong impression about distances and relative sizes. Different arrangements of the chart (known as projections) can achieve better understanding (Figure 2.37).

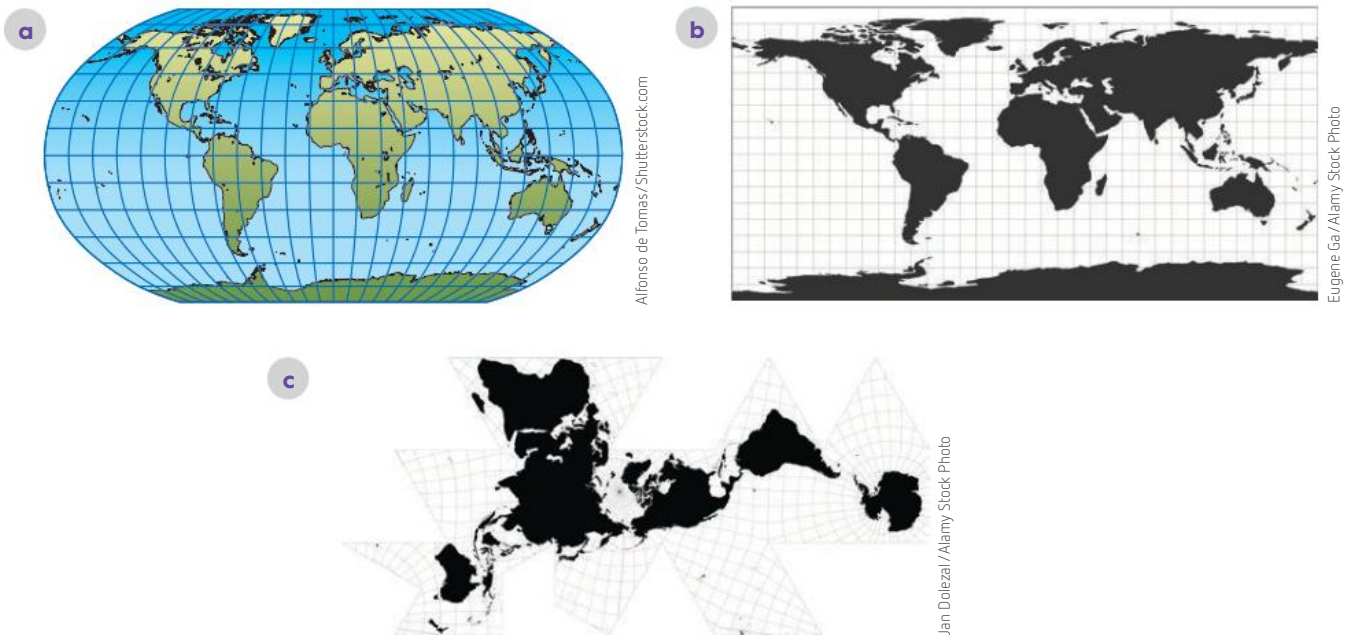


FIGURE 2.37 (a) A Robinson projection was designed to keep types of distortion (direction, distance, shape and size) low across the majority of the map. It becomes more distorted at the poles. Its purpose is to be visually appealing. (b) A plate carrée (flat square) projection has noticeable distortion. (c) A dymaxion projection has less distortion and there is no 'right way up'. This reduces the bias towards the Northern Hemisphere inherent in the other map types. However, it can be confusing to use.

Plot all the data

When summary calculations of data are made, the details are lost. The various forms of averages (mean, mode, median, variance, standard deviation) all calculate a central 'representative' value. If a trend line or average is the only information quoted, then significant details may be disguised or obscured and the result may just be misleading.

The famous example by Frank Anscombe illustrates the significance of seeing all the data (Figure 2.38). The descriptive statistics, mean, variance and regression trend line are very similar – to the point of being identical – yet the data sets are completely different.

In the hands of novice users, statistical packages can produce many meaningless statistics. Mathematicians would know that trend lines do not apply in every case. The use of trend lines is not appropriate for non-linear relationships and outliers.

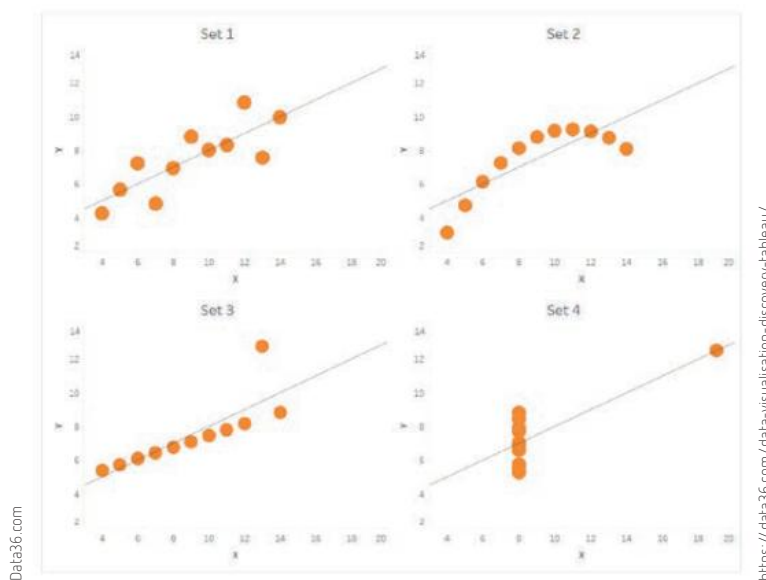


FIGURE 2.38 The descriptive statistics (mean = 9, variance = 3.32) and regression trend lines are identical for all four data sets graphed. When the data is plotted as a scatter plot, you can see that the data sets are completely different.



Weblink
How couples met

Animation can show development over time

On-screen visualisations have a clear advantage over static or print. See the weblink ‘How couples met’ to see the relative frequencies of the manner in which future couples first met over the decades from the 1960s to the 2010s. In Figure 2.39, you can see the movement of the *Met online* category from nearly last in 1960 to number one in 2010.

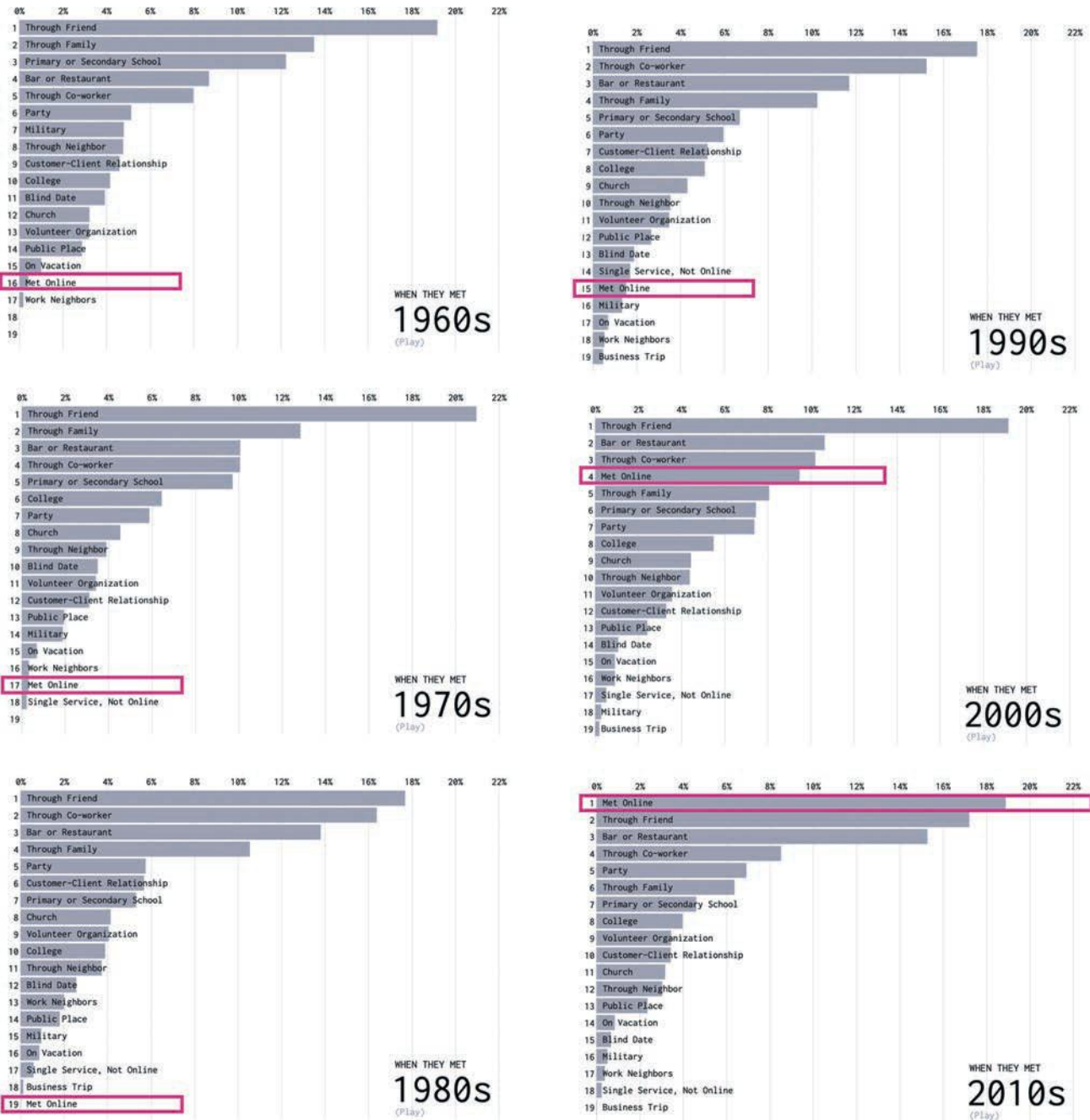


FIGURE 2.39 How couples met, 1960–2017

Dynamic data visualisations

A **dynamic data visualisation** is one where the **appearance** of the infographic can be changed (see Figure 2.41). This can be done by the user selecting alternative views, which may have been previously processed; in this case, it is also known as an interactive visualisation. Changing the appearance of the infographic can also be achieved by updating the data, or by the data being continually updated so the latest information can be included in the presentation.

Immersive media is becoming more widespread and acceptable when providing a 'lived' experience. The viewer, rather than observing a 2D or flat image, can get in between the data points in a virtual reality (VR) 360° data model, 3D or 360° graphs, which can be seen through a VR viewer.

Guidelines for visualisation components

You should follow some simple rules with visualisation components:

- They must have titles.
- The x-axis and the y-axis must be labelled.
- Use a key (or legend) if more than one set of data is provided on the same graph.
- Include author identification and/or the source of the data, the date and a filename (if appropriate).
- Include the unit of measurement on the relevant axis.
- Label each segment of a pie chart.
- Arrange segments of a pie chart (starting at 12 o'clock position) from smallest to largest.
- Consider including absolute figures as well as percentages.
- Choose colours that match the information being discussed.
- Use graphs to show trends or relationships between values on each axis.
- With pie charts, if possible, limit the number of items represented in a chart to five or six.

2.3 THINK ABOUT APPLIED COMPUTING

- 1 Search the web for the latest VR/AR data visualisations.
- 2 Go to the Infogram website to view several examples of VR data visualisations. How might VR graphics be more effective than 2D graphics?



Weblink
Infogram



FIGURE 2.40 Geospatial data can be used to create 3D visualisations that multiple users can view through a VR viewer.

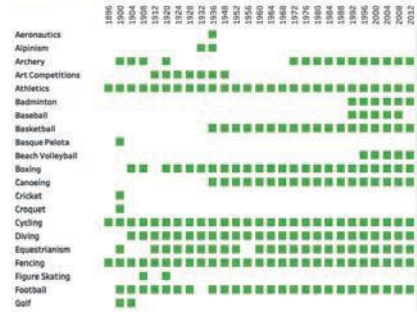
A Brief History of Olympic Sports

Click on a sport to see medal results

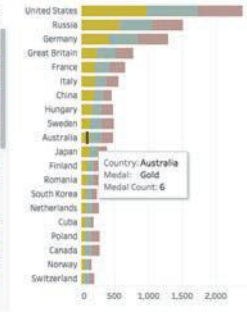
Pictograms



Olympic Sports



Sport: All, Games: All



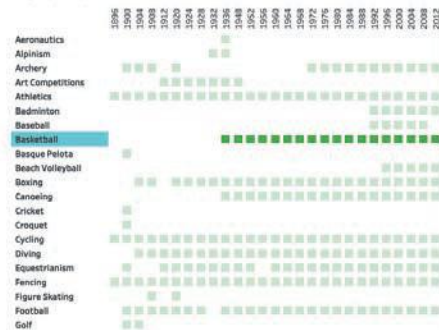
A Brief History of Olympic Sports

Click on a sport to see medal results

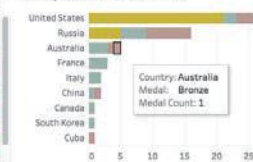
Pictograms



Olympic Sports



Sport: Basketball, Games: 1936, 1948, 1952 and 15 more



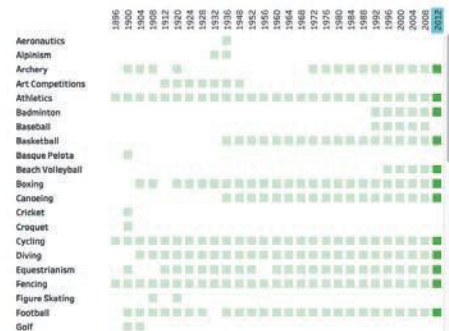
A Brief History of Olympic Sports

Click on a sport to see medal results

Pictograms



Olympic Sports



Sport: Archery, Athletics, Badminton and 29 more, Games: 2012

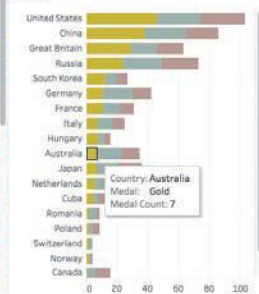


FIGURE 2.41 Dynamic data visualisations have a menu that allows choice of data to be viewed. Interactive features included: Tooltip hover, Menu buttons, Year button, Country button.

Tableau.com

Tableau.com

Tableau.com

Choose and use your software carefully

The software chosen to create your visualisation components will be capable of reading, processing and displaying your data set. Some data in raw form needs to be ‘cleaned’ or wrangled in order for the chart to be displayed. For example, some elements may have text instead of numbers recorded, or have empty fields. Sometimes what appears as numbers has been entered as text (or string) and will need to be converted. If the raw data is presented in a table format (rows and columns), the headings will need to be verified so each of the column data is known to be as it has been assumed. A chart can be created once all the data has been verified and cleaned ready for processing. Microsoft Excel, Tableau and Wolfram Mathematica are readily available software applications that can provide tools to construct many types of charts. These applications are downloaded and installed to a single computer. Online tools such as Plotly and DE3 require registration to save the graphic. Combinations of D3.js, R and Python would be considered industry standard; however, these are well beyond expected VCE skills and knowledge.

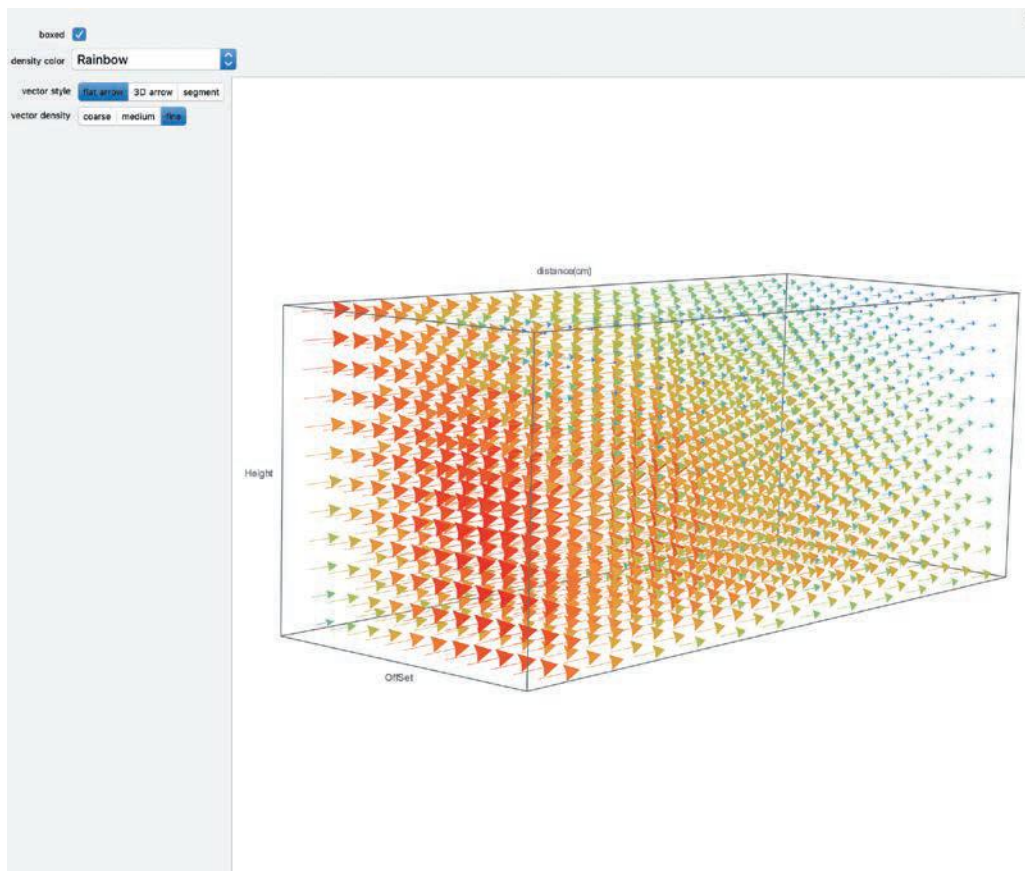


FIGURE 2.42 A Mathematica interactive data visualisation showing air speed near a circular fan (90°) in a model wind tunnel. The air speed is fastest near the fan blades, slowest at the centre and edges. Menu choices alter the view displayed.

It is important to clearly define and understand the functional and non-functional requirements, constraints and scope when planning a data visualisation.



CASE STUDY

Smart meter data

Victorian electricity subscribers can apply to their provider to obtain a record of their electricity consumption. Smart meters installed at most premises record consumption every 30 minutes. Consumers can analyse their consumption patterns and consider whether to change their electricity use behaviour.

Many consumers have adopted a time-of-use electricity plan. Certain 'peak' and 'off-peak' time periods are designed through the day. The pricing is higher or lower at these times to encourage or discourage consumption. Note: every retailer has a different pricing structure so making comparisons becomes very difficult, unless you have access to data visualisation software.

Several retailers have seen the confusion deliberately created by many of the existing retail players and have provided 'real time' calculation of consumption and pricing to gain a marketing edge.

20170101	0.130	0.145	0.15	0.157	0.139	0.148	0.143	0.157	0.139	0.15	0.149	0.139	0.074	0.015	0	0	0.031	0.034	0.005	0.006	0	0	0	0	0	0	0	0	0	0	0.004	0.185	0.204	0.318	0.476	0.352	0.562	0												
20170102	0.214	0.189	0.174	0.138	0.136	0.125	0.132	0.123	0.119	0.127	0.204	0.101	0.140	0.002	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.015	0	0.026	0.336	0.243	0.251	0.141	0.138	0											
20170103	0.11	0.13	0.102	0.089	0.121	0.125	0.118	0.109	0.111	0.127	0.15	0.249	0.002	0.073	0.021	0.006	0	0	0	0.042	0	0	0	0	0	0.026	0	0	0	0	0.02	0	0.024	0.011	0.202	0.223	0.467	0.461	0.331	0.289	0									
20170104	0.149	0.17	0.141	0.135	0.122	0.145	0.141	0.137	0.137	0.139	0.116	0.147	0.004	0.022	0	0	0.079	0.429	0.072	0.223	0.444	0.481	0.396	0.007	0.003	0.007	0.09	0.129	0.106	0.140	0.108	0.204	0.004	0.709	1.527	0	0	0	0	0										
20170105	0.14	0.175	0.151	0.138	0.131	0.129	0.116	0.114	0.11	0.114	0.116	0.044	0.073	0.016	0.276	0.23	0.044	0.015	0.03	0.015	0.107	0.307	0.306	0.136	0.204	0.207	0	0	0.007	0	0	0.004	0.042	0.004	0.313	0.191	0.117	0.308	0.143	0.138	0									
20170106	0.147	0.119	0.103	0.103	0.103	0.085	0.077	0.083	0.091	0.08	0.081	0.057	0.026	0.006	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.015	0.018	0.119	0.126	0.236	0.261	0.144	0.226	0										
20170107	0.121	0.136	0.102	0.111	0.147	0.123	0.105	0.107	0.112	0.12	0.109	0.23	0.004	0.002	0	0	0.02	0	0	0	0	0	0	0	0	0	0.015	0.023	0	0	0	0.047	0	0.081	0.118	0.126	0.236	0.261	0.144	0.226	0									
20170108	0.11	0.145	0.138	0.124	0.145	0.137	0.136	0.131	0.132	0.119	0.126	0.235	0.018	0.003	0	0	0.032	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.003	0.145	0.208	0.237	0.302	0.204	0.294	0	0										
20170109	0.142	0.111	0.121	0.104	0.131	0.118	0.113	0.116	0.117	0.119	0.071	0.144	0.104	0.025	0	0	0.003	0	0.107	0.033	0	0.046	0.118	0	0	0	0	0	0	0	0	0	0.201	0.281	0.244	0.119	0.171	0	0											
20170110	0.225	0.219	0.179	0.175	0.179	0.18	0.141	0.131	0.119	0.125	0.123	0.246	0.005	0.009	0	0	0	0	0	0.0208	0	0	0	0	0	0	0	0	0	0	0	0	0.033	0.013	0.174	0.007	0.078	0.135	0.147	0.171	0									
20170111	0.163	0.137	0.132	0.139	0.132	0.131	0.124	0.143	0.127	0.113	0.097	0.073	0.011	0	0	0	0.004	0.005	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.022	0.087	0.139	0.115	0.179	0.261	0.467	0.257	1.099	0								
20170112	0.17	0.181	0.165	0.161	0.165	0.172	0.167	0.178	0.176	0.144	0.124	0.139	0.083	0.006	0.08	0.023	0	0	0	0	0	0	0	0	0	0	0.046	0	0	0	0	0	0.051	1.017	0.082	1.009	0.084	1.151	0.163	0.413	0.32	0								
20170113	0.137	0.165	0.138	0.155	0.13	0.143	0.136	0.137	0.17	0.154	0.091	0.009	0.046	0.003	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.007	0.006	0	0	0	0.026	0.051	0.047	0									
20170114	0.234	0.171	0.111	0.108	0.209	0.209	0.234	0.199	0.191	0.157	0.108	0.234	0.142	0.139	0.047	0	0	0	0	0.027	0	0	0	0	0	0	0	0	0	0	0	0	0.013	0	0	0	0	0.003	0.145	0.208	0.237	0.302	0.204	0.294	0					
20170115	0.409	0.19	0.146	0.137	0.141	0.223	0.106	0.201	0.22	0.141	0.18	0.228	0.042	0.001	0	0	0.001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.147	0.004	0	0	0	0.022	0.009	0.258	0.208	0.157	0.304	0.477	0.472	0.507	0.301	0.409	0
20170116	0.16	0.153	0.141	0.162	0.141	0.149	0.149	0.134	0.147	0.141	0.13	0.136	0.07	0.116	0.109	0.006	0	0	0	0.053	0	0	0.073	0.04	0	0	0.025	0.07	0.072	0.041	0.122	0.074	0.111	0.15	0.209	0.119	0.211	0.408	0.412	0.247	0									
20170117	0.248	0.067	0.291	0.143	0.108	0.104	0.105	0.117	0.095	0.09	0.089	0.081	0.042	0.003	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.023	0	0	0	0	0.11	0	0.016	0.004	0.108	0.107	0.181	0.104	0				
20170118	0.208	0.279	0.142	0.12	0.113	0.119	0.139	0.128	0.114	0.113	0.123	0.118	0.001	0.002	0	0	0.044	0.024	0	0	0	0	0.078	0.002	0	0.004	0.119	0.022	0.004	0.12	0.113	0.044	0.145	0.276	0.444	0.402	0.59	0.727	0.765	0.621	0									
20170119	0.14	0.162	0.197	0.188	0.174	0.166	0.171	0.151	0.136	0.144	0.134	0.106	0.46	0.137	0.275	0.392	0.141	0.012	0.042	0.183	0	0	0	0	0	0.093	0	0.009	0.022	0	0	0	0	0.105	0.447	0.229	0.672	1.408	1.462	0.289	0									
20170120	0.142	0.161	0.142	0.146	0.131	0.14	0.142	0.14	0.138	0.139	0.144	0.147	0.179	0.246	0.154	0.146	0.246	0.202	0.7	0.073	0.033	0.036	0.446	0.007	0.229	0	0	0	0.073	0.107	0.161	0	0.022	0.003	0.241	0.216	0.246	0.491	0.304	0.288	0									
20170121	0.221	0.162	0.152	0.146	0.144	0.138	0.133	0.132	0.141	0.131	0.113	0.113	0.119	0.113	0.103	0.106	0.004	0	0	0	0	0.004	0.073	0.005	0.014	0	0	0.047	0.001	0.003	0.007	0.022	0.731	0.049	0.212	0.31	0.47	0.513	0.037	0										
20170122	0.243	0.105	0.107	0.161	0.14	0.137	0.135	0.134	0.127	0.137	0.134	0.124	0.221	0.076	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.137	1.043	1.171	0.89	0.554	0.555	0.704	0.438	0								
20170123	0.204	0.209	0.191	0.164	0.157	0.172	0.157	0.148	0.152	0.162	0.164	0.171	0.127	0.094	0.087	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.001	0.024	0.109	0.113	0.28	0.406	0.604	0.644	0.626	0.861	0.513	0						
20170124	0.157	0.146	0.148	0.144	0.136	0.143	0.137	0.14	0.132	0.144	0.137	0.139	0.149	0.049	0.036	0.031	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.001	0	0.024	0.011	0.209	0.603	0.607	0.417	0.517	0								
20170125	0.143	0.151	0.14	0.139	0.137	0.134	0.133	0.127	0.128	0.128	0.124	0.121	0.086	0.027	0.031	0.044	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.034	0.119	0.009	0.304	0.351	0.361	0.34	0	0									
20170126	0.192	0.167	0.165	0.158	0.166	0.161	0.167	0.1	0.19	0.164	0.153	0.164	0.167	0.051	0.037	0.06	0	0.119	0.042	0.14	0	0.014	0.001	0	0	0	0.275	0.017	0	0.001	0	0	0	0.036	0.205	0.418	0.619	0.617	0.504	0										
20170127	0.244	0.174	0.167	0.161	0.165	0.148	0.152	0.143	0.136	0.141	0.147	0.115	0.138	0.144	0.143	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.004	0.042	0.004	0.313	0.191	0.117	0.308	0.143	0.138	0							
20170128	0.137	0.18	0.18	0.18	0.144	0.146	0.134	0.136	0.148	0.125	0.127	0.137	0.14	0.1	0.183	0.007	0.033	0	0	0	0	0.007	0.438	0.031	0	0	0	0	0	0	0	0	0	0.028	0.207	0.308	0.767	0.622	0.861	0.861	0									
20170129	0.231	0.149	0.115	0.148	0.142	0.149	0.149	0.137	0.138	0.15	0.143	0.131	0.109	0.053	0.08	0.031	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.395	1.302	1.216	0.886	0.701	0.504	0.34	0.37	0								
20170130	0.183	0.239	0.176	0.178	0.176	0.166	0.176	0.167	0.159	0.164	0.162	0.223	0.1	0.099	0.104	0.017	0.006	0.113	0.056	0.049	0.042	0.056	0.029	0.02	0.024	0.007	0.032	0.009	0.072	0.06	0.013	0.046	0.252	0.331	0.246	0.391	0.384	0.252	0.265	0										
20170131	0.21	0.179	0.166	0.169	0.169	0.149	0.142	0.147	0.133	0.136	0.14	0.147	0.146	0.121	0.044	0.099	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.042	0.209	0.254	0.41	0.252	0.137	0	0									

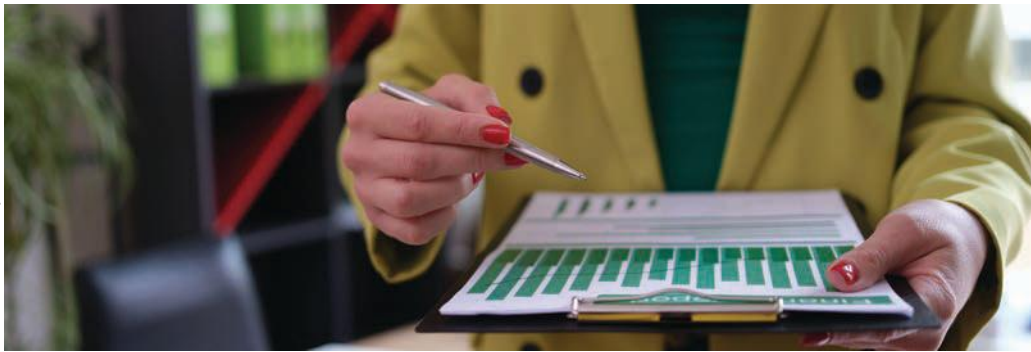
FIGURE 2.43 Raw data supplied by an electricity provider. Electricity consumption is recorded by the smart meter every 30 minutes.

Selina Dennis

Functional requirements

Functional requirements relate to what the data visualisation should do. Typically, functional requirements will describe a behaviour or function. This includes determining the input required, the desired output, data

Alexander Korch/Alamy Stock Photo

**FIGURE 2.44**

Visualisations can be used to show sales data.

Non-functional requirements

Non-functional requirements describe how the data visualisation appears rather than what it does. This requires consideration of usability, reliability, portability, robustness and maintainability.

Usability

All graphic solutions that represent data need to be easy to use. The user should be able to acquire the required information easily and efficiently. When designing your data visualisation for **usability**, consider the following:

- Can the user understand the graphic form and the information it conveys?
- Can the user quickly identify the purpose and interpret the critical data?
- Is the graphic form accurate and a true reflection of the data that has been analysed?
- Have you saved the final data visualisation in a format that maximises legibility, and in a size that will be quick to load?
- Have you chosen sensible dimensions? For example, you may want to think carefully before designing wider than most website body panes (around 800 px) for a browser presentation or 16:9 ratio for computer screens and (most) projectors.
- Have you chosen easily readable font sizes? Clear labels, legends and controls?
- Have you tested the data visualisation in multiple browsers to check that it works? If plugins are necessary, these must be included with instructions.



RESEARCH

Electricity consumption

Ask if you can get your household's electricity consumption file downloaded from your electricity supplier. Your school can also acquire this data. Alternatively, go to Nelson MindTap to download the sample file *SampleElectricity Consumption.csv*.

Explore the data to discover any 'dodgy' (unvalidated) data, which needs to be 'fixed' (corrected).

Unvalidated data is the contents of any cell that will not be plotted by software (i.e. not a number).

Step 1: Plot all the data

The only way to get an idea of what you have is to make a scatter plot of the data.

Step 2: Look for patterns

'What about ...?' is the key question to begin your investigation of the raw data. There may be 'odd' plots or **outliers**. These may be items of interest for further understanding. Each plot corresponds to a date and time. Peaks will indicate the largest amount of consumption. Individual dots may not show a pattern.

Step 3: Plot each series

On the same axes, plot each 24-hour day (48 readings on each of 365 days). Calculate the total consumption for each day.

Step 4: Plot each day as a bar chart

Alternatively, a stacked bar chart could show the consumption each day and the total each day. The tooltip hover feature will allow inspection of items of interest.

Note: For those consumers with solar PV panels, it will be possible to have 'negative' consumption indicated. Consider what this means for calculations. It will be necessary to consider those consequences for total consumption and for billing purposes.



Resource
Sample electricity
consumption

Reliability

Reliability involves the accuracy and consistency of the visualisation. This requirement ensures that the visualisation faithfully represents the underlying data without errors, providing users with trustworthy insights. Reliable visualisations are consistent across different uses and sessions, and they correctly update to reflect any changes in the data source.

Portability

Portability means that the visualisation should be accessible and function correctly across different devices, platforms and browsers. This ensures that users can view and interact with the visualisation regardless of the technology they are using. A portable visualisation adapts to different screen sizes and resolutions, providing a seamless experience whether accessed on a desktop, tablet or smartphone.

Robustness

Robustness relates to the visualisation's ability to handle unexpected inputs or large data sets without crashing or experiencing significant performance issues. A robust visualisation remains stable and responsive under various conditions, including high user traffic or complex data sets.

Maintainability

Maintainability refers to how easily the visualisation can be updated or modified to accommodate new data or changes in requirements. This ensures its long-term usability and relevance. A maintainable visualisation is built with clear, modular functionality and documentation, allowing developers to make necessary adjustments efficiently. This quality is essential for adapting to evolving data and user needs without requiring a complete redesign.

Constraints

Any factor that limits or restricts the data visualisation may be considered a **constraint**. Constraints can be broken down into categories: economic, technical, social, legal and usability.

Economic

Economic constraints include both time and budget. Either a lack of time or a small or restricted budget may result in a re-evaluation of the user's requirements, or a re-evaluation of how the requirements can be achieved. The deadline by which the user or client needs to have the solution operational will define the time available to design and develop the solution. The longer the time available, the more time there is to complete an in-depth analysis and detailed designs, and to develop advanced features of the solution.

Technical

Technical constraints are constraints related to the hardware and software available for the project. Available hardware and software, memory and storage capacity, processing and transmission speeds, and security concerns are all examples of possible technical constraints.

For example, the data collection may be incomplete, with gaps in the data sequence, or may have errors due to collection errors or transcription errors.

Chart size limitations

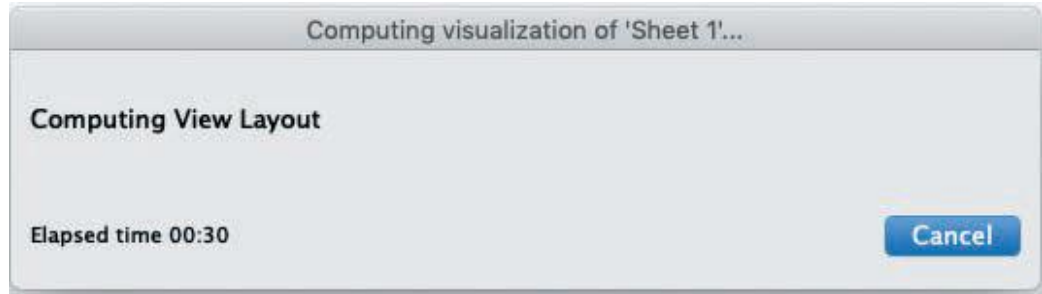
The width of any graphic will be approximately one screen, or 800px. The axes labels need to be chosen carefully to ensure clarity.

Software limitations

The computer hosting the data may not be able to manipulate, calculate or redraw quickly. After about five seconds, users begin to notice the computer is 'running slowly'. Big data may be defined as data sets that cannot be accommodated within RAM on host computers. The data sets accessed in this course are unlikely to approach that size. For example, in research and industry, data sets typically from 100 MB up to 1 GB file size may be considered small. Excel, for example, is difficult to use with file sizes greater than 10 GB.

Excel, which is used for data manipulations and some simple visualisations, is limited to 255 series and 16 variables (columns) and some number precision values. An Excel chart is essentially unlimited, with up to 1 048 576 rows by 16 384 columns and most other functions limited only by available computer memory. Tableau is used for data visualisation, and has limitations dependent on the host computer.

FIGURE 2.45 Tableau measured the time it took to refresh the display when rearranging more than 100 000 data points on a 4 GHz i7 CPU with 32 GB RAM. Other devices may be slower.



Selina Dennis

Non-technical constraints

Non-technical constraints relate to areas other than hardware and software: social, legal, usability. The user's level of expertise (social) is an example where a solution, when being developed for users with little digital systems expertise, may restrict some of the requirements that would involve complex manoeuvres to complete.

When considering usability, all visual solutions that represent data need to be easy to use, flexible and robust. The user should be able to acquire the required information easily and efficiently.

Legal requirements are another type of non-technical constraint. Privacy laws may restrict features linked to displaying personal data in the solution, or to collecting data from the devices of someone using your solution. Copyright laws may restrict features that allow other users to upload content to the solution.

Scope of a solution

The scope of a solution refers to the work that must be done to complete that solution. Scope will be informed by the functions and features of the solution. Scope is documented in a scope statement, which may be referred to later, in order to make decisions as the solution develops. Managing scope is about what is and is not included in the solution scope.

Design tools

Design tools are used to represent the functionality and appearance of your solution. Normally, the solution is first sketched on paper. Several design tools can be used to represent the design of graphic solutions; however, **annotated diagrams** are the most apt for planning a solution and presenting it.

Software such as spreadsheets or databases can assist with the manipulation of data to develop graphs and charts. This software has inbuilt search, sort and filter functions that enable users to work quickly and with minimal effort, thereby maximising **effectiveness**.

While there are a wide range of design tools that may be used to design the appearance of a solution, in VCE Applied Computing we shall consider:

→ appearance design tools

- » layout diagrams
- » storyboards
- » annotated diagrams and mock-ups

→ functionality design tools

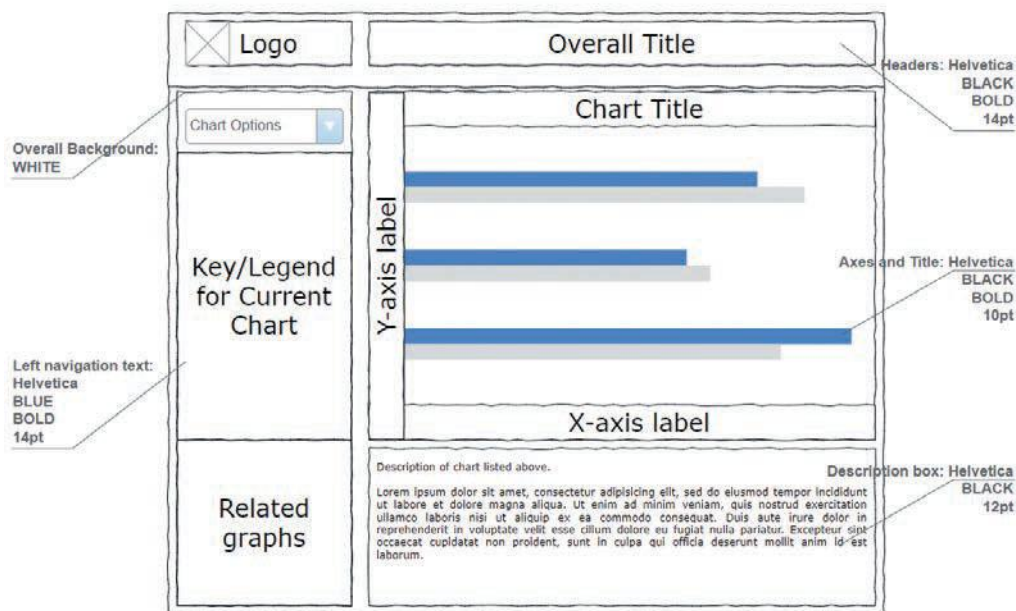
- » input–process–output (IPO) charts
- » query designs.

Appearance design tools

Annotated diagrams and mock-ups

Wire frames are annotated diagrams that provide a visual depiction of how your visualisation solution should look (Figure 2.46).

The designs indicate features such as differences in font size, colour and positioning of objects. The placement of objects must be planned so that a balanced, visually appealing and clear effect is attained.



THINK ABOUT 2.4 APPLIED COMPUTING

- 1 Hand draw one of the charts from this chapter as an annotated wire-frame diagram similar to Figure 2.46. (You will not be able to identify specific typefaces when you do this.)
- 2 Label the headers, axis titles, labels and chart type on the hand-drawn visualisation.

FIGURE 2.46 An example of an annotated diagram with a dashboard for viewing charts

The sketch is usually ‘rough’ in black and white, and contains structure and placement detail.

Mock-ups are usually full-colour, fully featured ‘fakes’ that give an indication of what the final visualisation may look like.

What is the difference between a wire frame and a mock-up?

A wire frame is typically a rough sketch drawn in a single colour that provides layout information: location, spacings, and content details listing colour, style, font and size (if known).

A mock-up provides format information. Colour, size, font and style details are shown on a full-colour ‘fake’ or facsimile that creates a ‘look and feel’ of the final visualisation solution.

The annotated diagram should clearly indicate:

- the shape or type of graphic solutions
- titles and layout of the diagram
- formats and conventions that are used in graphic solutions.

Before using an annotated diagram, you should consider the data set that will be used as the basis of the graphic solution. The amount and type of data, and purpose for developing a graphic solution of the data, will help you decide on the type of graphic solution to create.

Functionality design tools

IPO charts

An **IPO chart** (Table 2.13) is used during the design stage to clearly identify the solution’s input and output, and the processing steps required to transform the data into information. By completing an IPO chart, the developer gets a sense of how much formula development work might occur during the manipulation stage of the project. An IPO chart can be used to show how data is processed into meaningful information. An IPO chart, also called a ‘defining diagram’, identifies what data is required for the solution (input), what information the solution needs to produce (output), and the processing manipulation activities required to transform the data into information, or the function of the solution. In the case of creating a graphical solution, the input requires an understanding of what data is needed and where it is coming from, the process focuses on the functionality of the solution and the output refers to the graphical solution that is being created.

TABLE 2.13 An example of an IPO chart for investigation into energy consumption

Input (data)	Processing (calculations)	Output (information)
Daily 30-minute readings for energy consumption	Tally the value for each day	Daily total
Weekly energy consumption	Tally the results for each week 1–52 for given year	Weekly total
Monthly energy consumption	Tally the results for each month	Monthly total
Amount of solar energy generated daily	Tally the negative values	PV solar energy
Energy bill for each month	Tally the energy bill using peak (3 p.m.–9 p.m.)/off-peak tariff (other times)	Energy bill totals
Reminder: Do not include PV solar values		

Query designs

When using SQL, various query designs can be planned to address different data retrieval, manipulation and analysis needs. Table 2.14 shows some common types of SQL query designs.

TABLE 2.14 Common types of SQL queries

Type	Description	Example
Select queries	Simple select queries are used to retrieve data from one or more tables. They can include filters, sorting and basic aggregations.	<pre>SELECT first_name, last_name, email FROM employees WHERE department = 'Sales' ORDER BY last_name;</pre>
Join queries	Join queries combine rows from two or more tables based on a related column between them. Types of joins include INNER JOIN, LEFT JOIN, RIGHT JOIN and FULL OUTER JOIN.	<pre>SELECT e.first_name, e.last_name, d.department_name FROM employees e INNER JOIN departments d ON e.department_id = d.department_id;</pre>
Aggregate queries	Aggregate queries use functions like COUNT, SUM, AVG, MIN and MAX to perform calculations on a set of values, often combined with the GROUP BY clause.	<pre>SELECT department, COUNT(*) as employee_count, AVG(salary) as average_salary FROM employees GROUP BY department;</pre>
Insert queries	Insert queries add new rows to a table.	<pre>INSERT INTO employees (first_name, last_name, email, department_id, salary) VALUES ('Seamus', 'H', 'seamus.h@example.com', 3, 60000);</pre>
Update queries	Update queries modify existing data in a table.	<pre>UPDATE employees SET salary = salary * 1.05 WHERE department_id = 3;</pre>
Delete queries	Delete queries remove rows from a table.	<pre>DELETE FROM employees WHERE department_id = 5;</pre>

Using temporary guidelines

When planning your solution, manage the elements of your data visualisation by drawing a visual 'grid' (Figure 2.47). This will help you to work out where you can place items in your solution, how much space you have, and what kind of **alignment** works best for each element.

This approach can be applied to database input forms and reports, and to spreadsheet layout(s) for workbooks with multiple worksheets, as well as data visualisation graphic solutions, which often have multiple charts and text combined in a single dashboard screen.

Figure 2.47 is a generic visual grid that can be adapted for database, spreadsheet and visualisation designs. By adopting a template that can be repeatedly applied, you introduce consistency into the look of your solutions, and improve your **efficiency** by reducing the time taken in arranging and rearranging objects for your solutions.

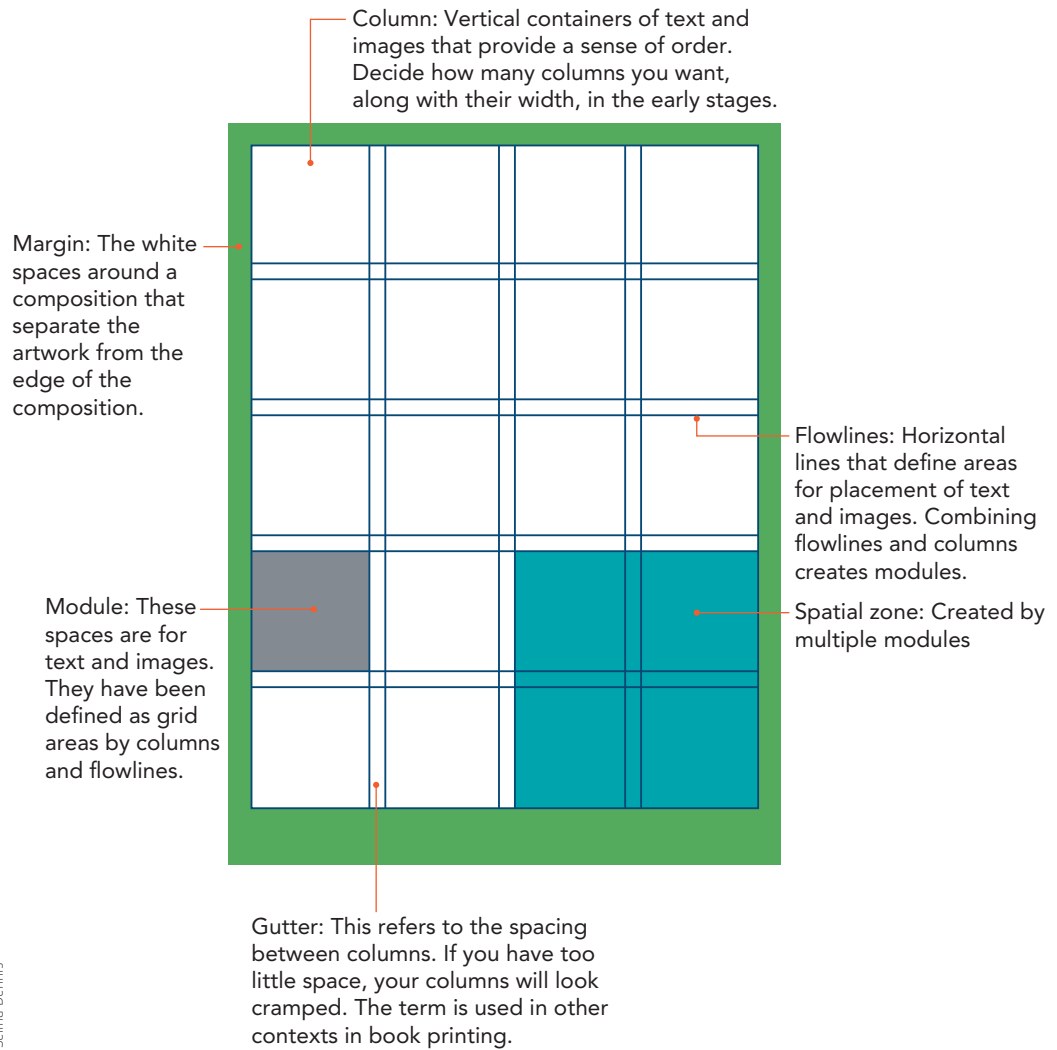
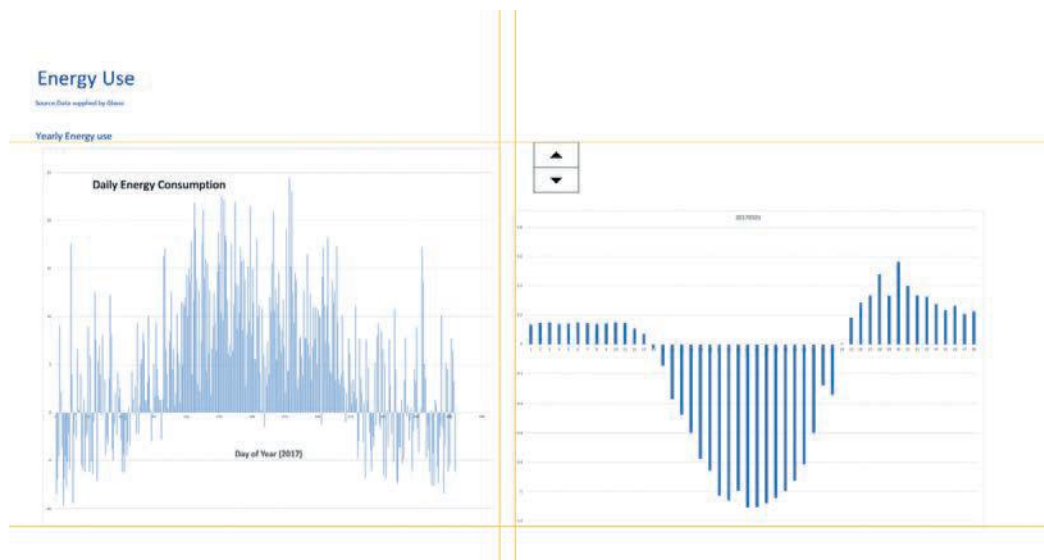


FIGURE 2.47 The visual grid will not just help you with alignment; it also helps with balance, space and accessibility.

Selina Dennis

FIGURE 2.48 Imagine the layout of your visualisation with a grid overlay to align each element. Heading, label, legend/key and axes can have their priority changed by placement or text emphasis (size, colour, style).

Data: Gbass



Formats and conventions

Formats are related to the font type and size selected, background colour used and any other change of appearance. **Conventions** are general rules that are followed when using a particular format. When we present data in a table, often the column headings are in bold, the text left aligned and the numbers right-aligned. When presenting data in a website, we use clearly named hyperlinks to make it easy for the user to find the required information. When addressing a letter, there are standard rules (or conventions) which should be followed. For example, the stamp goes in the upper right-hand corner, the return address goes in the upper left-hand corner while the 'to' address is centred on the envelope (Figure 2.49).

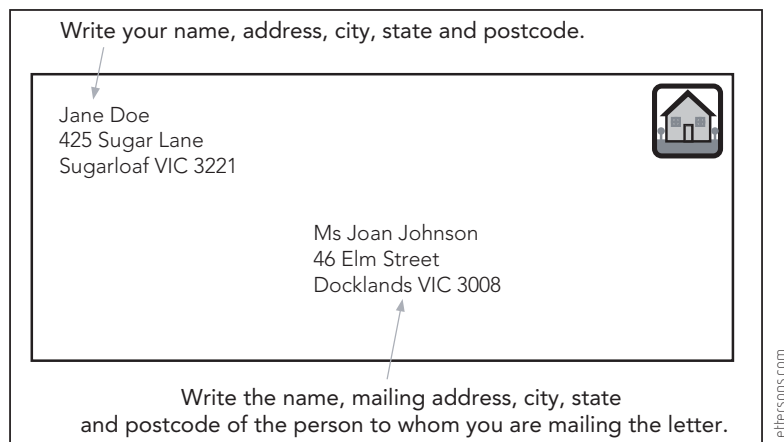


FIGURE 2.49 Conventions followed when addressing an envelope

With data visualisations appearing in so many different formats, there are a range of conventions that would apply to each format.

Some common conventions that would improve the effectiveness of the visualisation would include:

- clearly title the visualisation and explain its purpose
- label axes if appropriate
- use a key or legend to identify different data items
- include the name of the author and the source of the data
- identify the units of measurement
- choose colours that match the information being discussed.

Filenaming

The International Organization for Standardization (ISO) established date formats in 1988. These references can be included in filenames – for example, 20200407dataVisVer2.xlsx.

- formats and conventions suitable for databases, spreadsheets and data visualisations, such as:
 - » consistent naming conventions for databases (database name, table name, column name, primary key, foreign key)
 - » consistent naming conventions for spreadsheets (worksheet names, header labels for rows, header labels for columns)
 - » use of colours, fonts, images and icons.

2.5 THINK ABOUT APPLIED COMPUTING

Study a number of data visualisations you find effective. Try to identify the formats and conventions that have been used to make the solution stand out.

ISO 8601 date standard specifies dates should be represented in the form YYYY-MM-DD.

The hyphens can be omitted to use space more efficiently. For example, 2 April 2020 becomes 20200402.

Time can also be represented in a similar way using HH-MM-SS. Hours are represented using a 24-hour clock. For example, 2:34 p.m. becomes 1434.

Dates and times in these formats are easily comparable and can be sorted.

Databases

Establishing consistent naming conventions for databases helps to maintain clarity, ease of understanding and efficient management.

- Database name: descriptive enough to convey the purpose or content of the database, making it easy to identify its use and scope within the broader system.
- Table name: should reflect the entity or collection of entities the table represents, providing a clear indication of the data stored within it.
- Column name: should describe the data stored in the column, ensuring that it is immediately understandable to anyone reading the schema.
- Primary key: should uniquely identify each row in a table, helping to facilitate the unique identification of records within the table.
- Foreign key: should reference the primary key in another table, establishing and clarifying the relationships between different tables in the database.

Table 2.15 lists some suggestions for naming conventions across various database components.

TABLE 2.15 Database naming conventions

Component	Suggestions
Database	Use singular nouns (e.g. EmployeeManagement, SalesDB). Avoid special characters or spaces. Include version/environment if relevant (SalesDB_v1, SalesDB_Prod).
Table	Use plural nouns (e.g. Employees, Orders). Avoid using reserved SQL keywords. Maintain consistency across all tables.
Column	Use meaningful, descriptive names (FirstName, OrderDate). Prefix with table name or entity for clarity in joins (employeeID, orderDate).
Primary key	Use ID or table-specific identifier (EmployeeID, OrderID). Maintain consistency (ID or id for all primary keys).
Foreign key	Include the name of the referenced table followed by ID (EmployeeID, ProductID). Ensure clarity in relationships (parentID, childID)

Spreadsheets

Consistent naming conventions in spreadsheets make it easier for users to understand and manage data. They improve collaboration by ensuring everyone interprets the structure in the same way and also help to maintain data integrity by reducing errors. Additionally, consistent naming presents a professional, organised appearance when sharing spreadsheets with others.

- Worksheet names: should be descriptive and clearly indicate the content or purpose of the worksheet. This helps users quickly understand and navigate through the workbook.
- Header labels for rows: should describe the type of data contained in each row. Clear and descriptive labels make it easier to understand the structure and content of the data.
- Header labels for columns: should specify the type of data contained in each column, making it easier to know what type of data to enter. It also assists analysis and reporting by providing clarity.

Table 2.16 lists some suggestions for naming conventions across spreadsheet components.

TABLE 2.16 Spreadsheet naming conventions

Component	Suggestion
Worksheet names	Use meaningful and descriptive names (Sales2025, EmployeeData). Keep names concise but informative. Avoid special characters and spaces.
Header labels for rows	Use singular nouns that describe the data type (Product, Region). Ensure labels are concise and clear. Use consistent formatting, such as camel case or spaces between words (Product Name, Region).
Header labels for columns	Use descriptive labels that clearly define the data in the column (FirstName, OrderDate). Keep labels concise but informative. Use camel case, underscores or spaces for readability (First Name, Order_Date).

Data visualisations

In addition to the basic design principles you should follow when creating your data visualisation, there are a number of useful formats and conventions suitable for graphic solutions, such as titles, text styles, lines and arrows, sources of data and legend, and colours and contrasts.

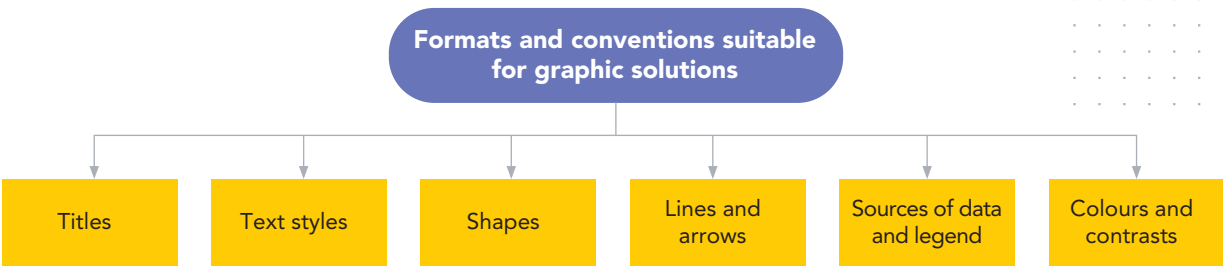


FIGURE 2.50 Formats and conventions

Titles

In the simplest of terms, adding a title to a document makes it a dominant element. Titles are generally styled as headings, with type that is bold and larger than the body text or subheadings. Titles make an impression. Titles should be concise, to-the-point and easy to say. Your title should be in larger text than the rest of your solution – perhaps at least 20pt if the body text is 10pt. Location could be left justified at the top, or centred for greatest emphasis. Bear in mind the F-shaped reading patterns of uncommitted readers.

Text styles

When we discuss text styles, we are essentially talking about fonts. A font is a typeface (such as Times New Roman, Arial or Calibri) plus its attributes (such as 20pt, bold, red). You may already know a few standard, familiar typefaces, such as:

- Times New Roman, a serif typeface. Serif typefaces have tiny marks or ‘tails’ on the end of the horizontal and vertical strokes of each letter. Serifs are used in books for body text and especially for long passages of text. This paragraph is in a serif font.
- Arial, a sans serif typeface. Sans serifs do not have the serifs on the strokes of each letter. They work best for short paragraphs, large headings and online, but not for long passages of printed text. The ‘Think about Applied Computing’ box (right) is in a sans serif font.

2.6

THINK ABOUT
APPLIED COMPUTING

Compare these dates and times using local AU/UK, US, JP formats and ISO8601:

US format: MM/DD/YY

AU/UK format: DD/MM/YY

Japanese format: YY/MM/DD

a 20th of November 2021

b 11th of February 2020

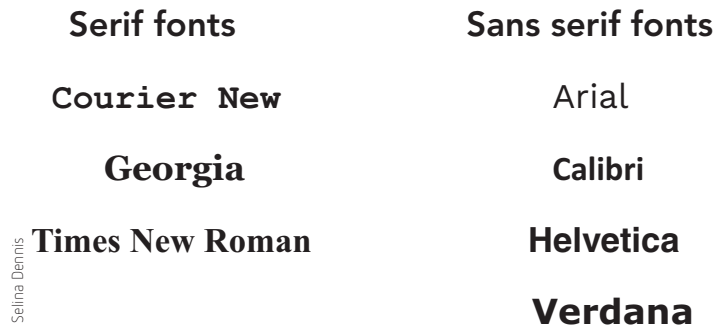
c 11:59 p.m.

d 12 a.m.

e 12 p.m.

FIGURE 2.51 Fonts with and without (sans) a serif offer different emphasis and may be suitable for titles, axis labels, value labels and digits. Generally, sans serif fonts are clearer for readability.

→ Courier New, a slab serif typeface, often used in programming. Slab serifs are best used when the focus is on function and not appearance. Slab serif fonts ensure that your characters are legible and unmistakable.



There are other typeface styles, such as handwriting, script and decorative. You will know decorative fonts such as Impact because it is the typeface predominantly used in memes online. Comic Sans, a casual script or handwriting typeface, is also well known.

When you are choosing text styles for your graphic solution, keep things simple. Use a few well-chosen typefaces, perhaps three at most, and use bold, italic, colour and point size to set out heading levels and distinguish between different types of text.

Text styles will apply some contrast while promoting a streamlined, professional appearance. However, using many different typefaces in one space can be untidy and overwhelming.

Remember: less is still more – bigger is not always better. Really think about what needs to be emphasised the most and what needs to be highlighted. Not everything needs to be bold, italic and 40 pt.

Lines and arrows

A line is a versatile visual element that uses only length and width. Lines can be:

Solid	
Dashed	
Dotted	
Broken	
Double	
Thick	
Thin	
Curved	
Freeform	

→ Bold or thick lines work well for emphasis and for representing a structure within a space. The thicker the line, the more it will draw the eye to the space, but the more crowded and boxed in it will look, so use a thick line carefully.

→ Light and fine lines can suggest technical details but also retain a sense of minimalism. Line thicknesses you might use range from:



You can use lines in your visualisation as borders or containers for sections of text, images or logos.

Charts

While there are very few mandatory rules, there are some strong recommendations for effective charts:

- Simplicity is preferred over complexity.
- Use colour tone rather than colour variety.
- Label axes with meaningful names, include units or denominations.
- Careful choice of labelling may reduce the need for legends and keys.
- In most cases, begin value axes at zero.
- Cite the sources of the data on the chart.

2.7 THINK ABOUT APPLIED COMPUTING

- 1 Identify the differences between **Pascal case**, **camel case** and **Hungarian case** naming conventions.
- 2 Hungarian case is no longer recommended when hyperlinks are used. Why not?
- 3 Spaces are discouraged in filenames and variable names. Why?

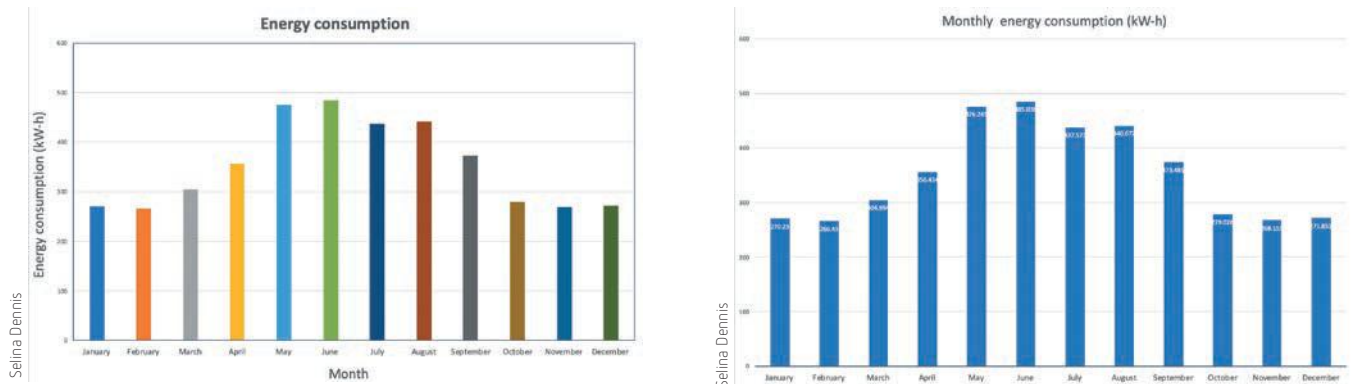


FIGURE 2.52 Simplify charts by reducing borders, gridlines and legends. Labelling is often duplicated without improving understanding – for example, different colours for different months when the months are already labelled.

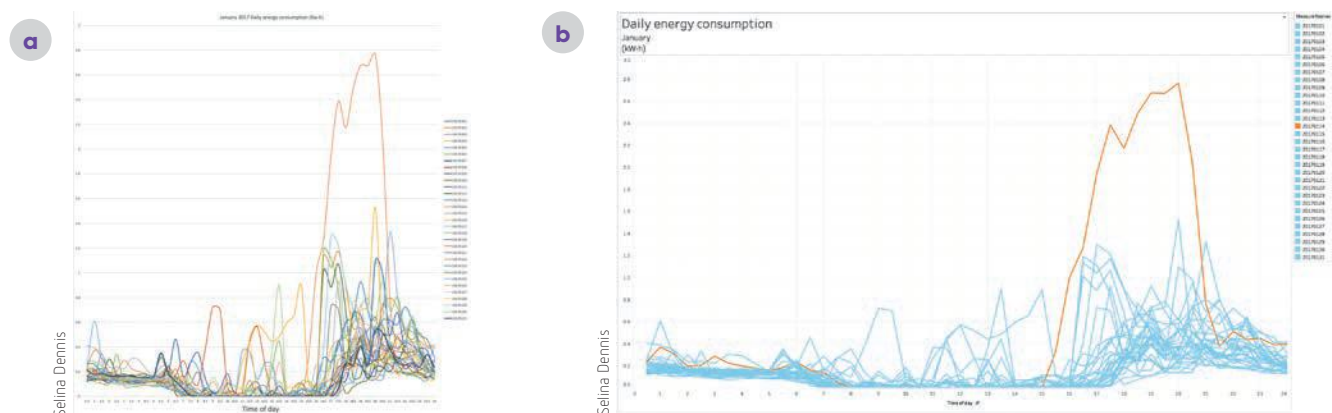


FIGURE 2.53 Line charts showing daily changes for January. (a) Excel default showing a colour for each of 31 days, individual colour highlight can be achieved using Developer Controls. (b) Tableau allows simple choice of day to highlight. Both allow tooltip hover to provide specific values for data points.

- If choosing a pie chart, for less than six categories, clearly label each sector with name and value. Ensure that the pieces add to 100 per cent.
- If choosing bubble chart comparisons, it is the area that is the comparison value, rather than the diameter.
- When plotting all the data, ensure that there is clarity by highlighting the focus of interest.

Data visualisation software

There are several options for data visualisation software. Excel, Tableau, Google Studio and Plotly all provide adequate charting opportunities to develop data visualisations and reporting dashboards.

Validation in this context is being done after the data has been recorded, captured and stored as a table of records or in database records. For visualisation software to deal with data that is unacceptable for display due to being out of range, and empty (or null) field, the wrong data type or in a different or incompatible unit, a filter will be applied.

A filter will cause empty or null fields to not be represented, rather than be plotted as a zero value. It is not acceptable to assign a zero value. Zero is still a value, when the data had no value. These null values must be found, rather than processed and permitted to create phantom data points, which appear to be valid.

Often the only way to discover such anomalous behaviour is to perform some trials to observe the patterns. The data sets are frequently so large that visual inspection is not possible or productive. One technique to locate anomalous data is to conduct a histogram sort and observe the distribution of data values. This may identify unusual outlier values, which may not be empty (null) or zero and just data entry errors.

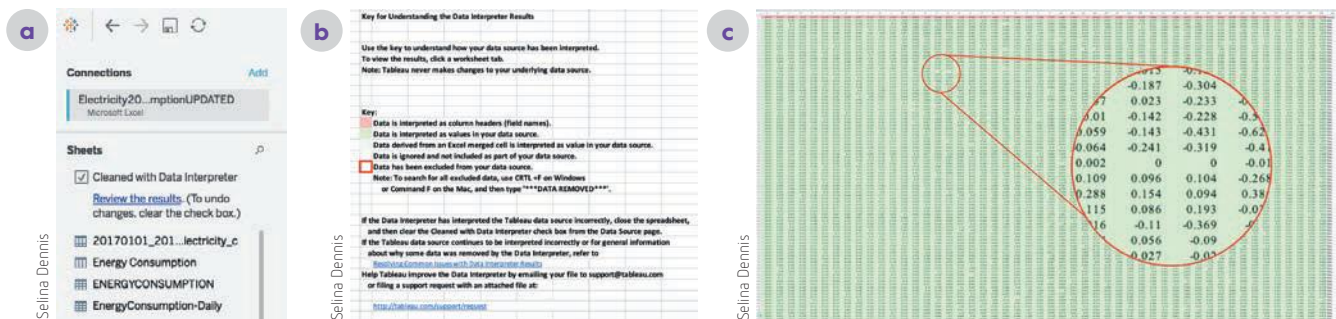


FIGURE 2.54 Tableau has a data interpreter function that removes empty/null values and 'cleans' the data in preparation for visualisation.

A further filter function allows any field to be inspected for both range and null values.

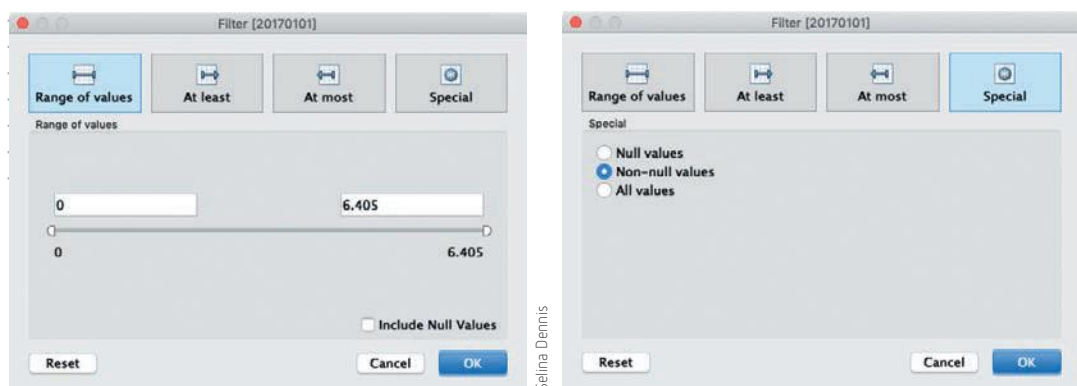


FIGURE 2.55 Tableau data filter allows a nominated field to be checked for within range compliance and other value outlier behaviour.

Interactive charts

Excel and Tableau have controls that can alter settings so the chart display changes on the one screen. Data and calculations reside on separate sheets. Control settings allow choice of variables to show a chosen slice of the data set.

Excel has **Form Controls** that can be inserted into a sheet and will provide user choice of the chart and data displayed.

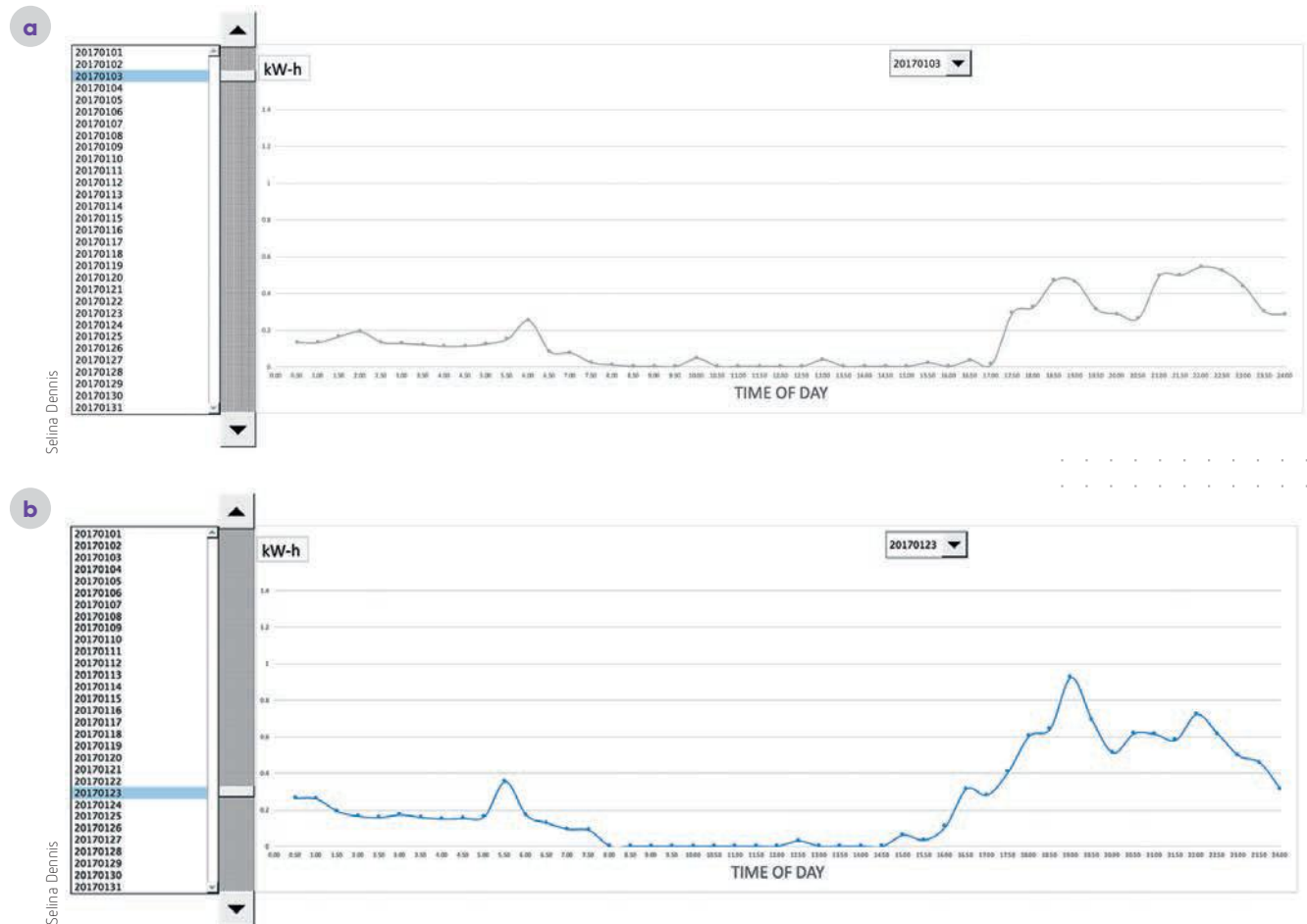


FIGURE 2.56 Excel charts controlled by Combo Box, Scroll Bar (slider) and List Box (drop-down menu)

There are several choices for Form Controls.

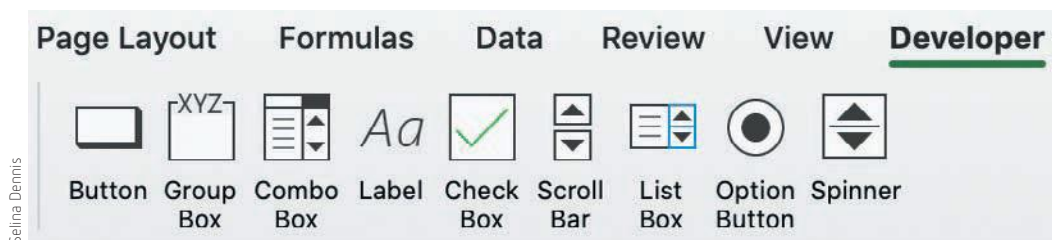


FIGURE 2.57 Excel has several Developer Tools Form Controls, which provide user control of the chart or sheet.

A clear benefit of using Form Controls to validate data entry is the reduction in typing errors. However, wrong choices from the displayed menu items may still occur.



CASE STUDY

Energy Superpowers

In 2018 Monash Tech School introduced a Year 8 Energy Superpowers program. This interactive program explores six energy usage profiles for the similarities and differences on electrical energy consumption and costs.

The entire program is written in the Wolfram Language using Mathematica as the application. The source data is held in a single Excel workbook with six worksheets with identical layout.

Privacy considerations

The data was provided to consumers at no cost by their energy providers. Monash Tech School acquired the data on the strict undertaking that the personal details should not be revealed except in the broadest sense to assist with matching energy use with user profile behaviour.

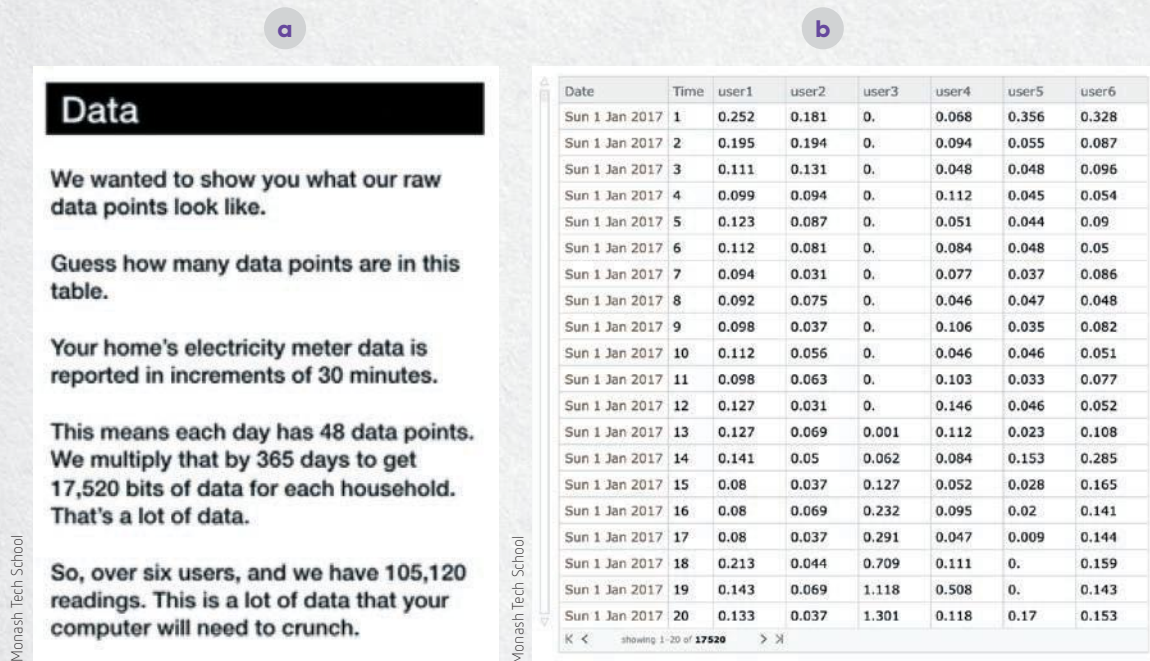


FIGURE 2.58 Data processed by the program has more than 105 000 data points.

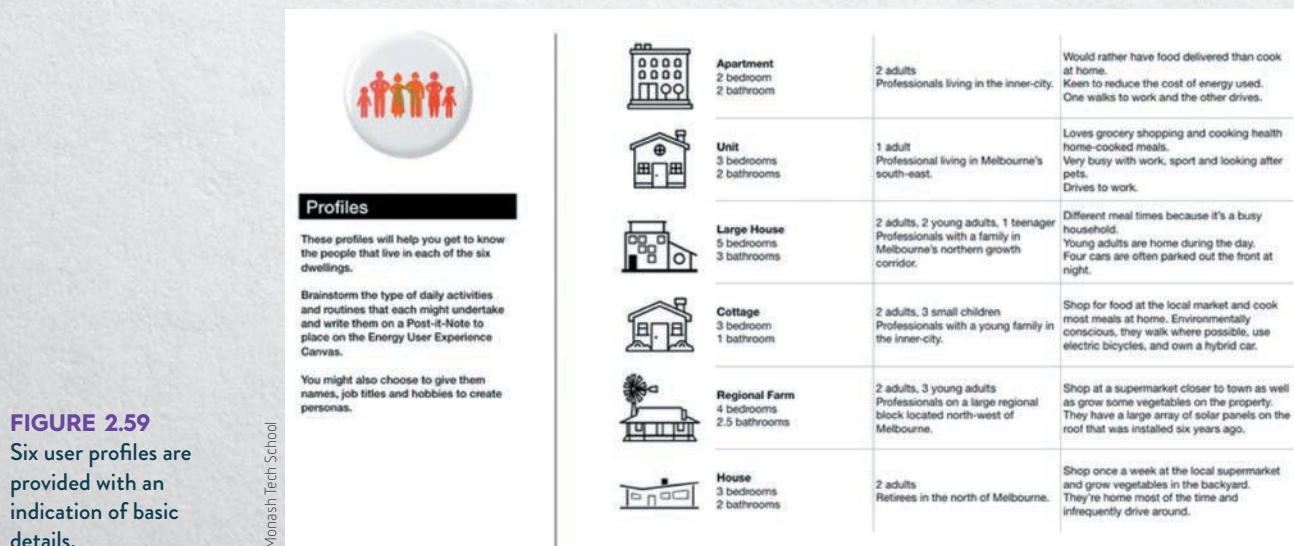


FIGURE 2.59 Six user profiles are provided with an indication of basic details.

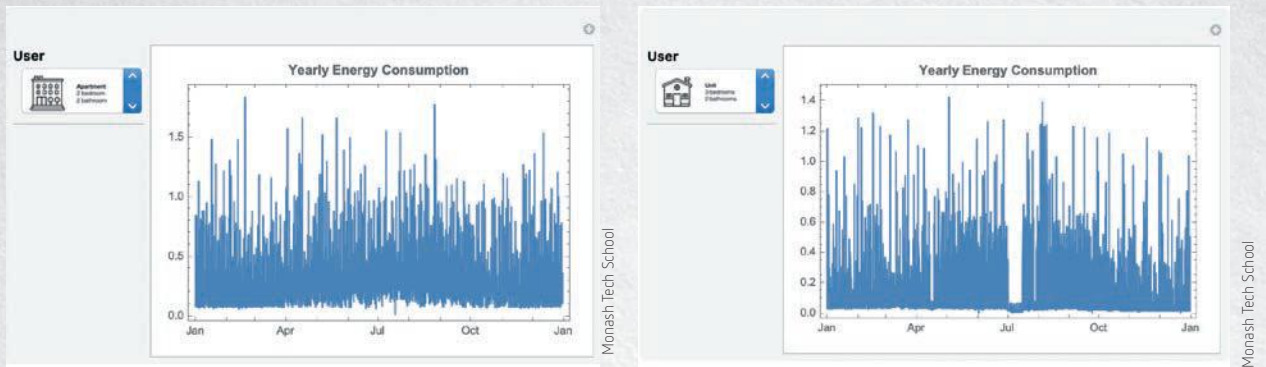


FIGURE 2.60 User energy consumption shows certain user behaviour and holiday periods.

When plotted in 3D, daily and monthly patterns across the year can be observed. The object can be rotated to allow identification of key points and daily, weekly and monthly activities.

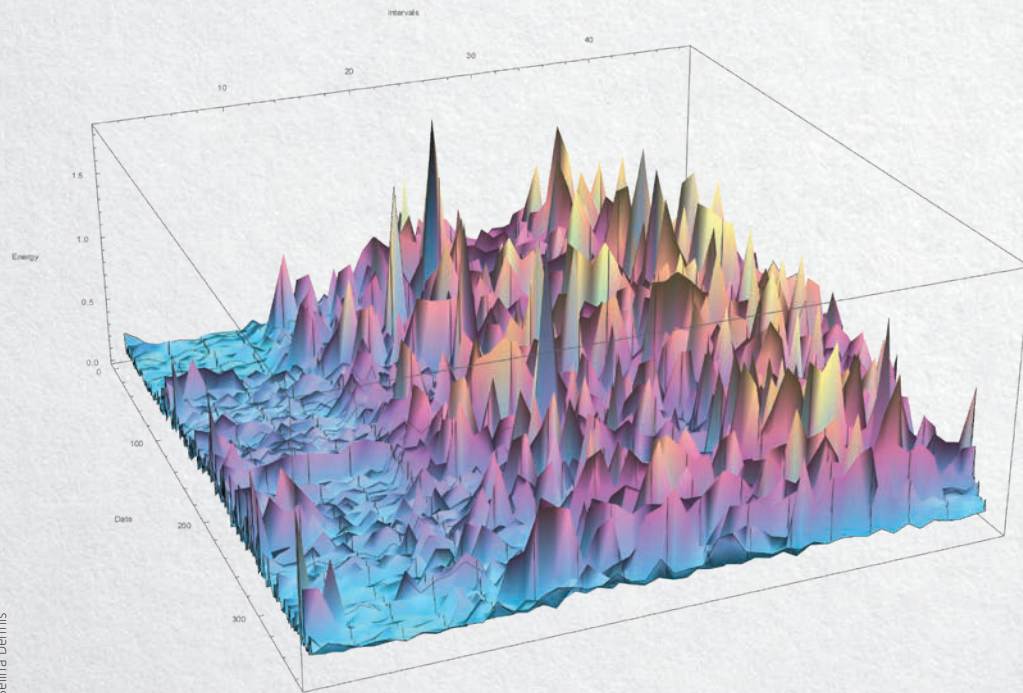


FIGURE 2.61 User 1, yearly energy consumption for each interval of 30 minutes for each day (17 520 data points)

So far, each chart is displaying raw data arranged by different time intervals. As 17 520 data points are too many to consider individually, it is much better to consider summary data.

Monthly billing is a usual summary value of what retailers charge for their services. Consumers have the opportunity to modify their energy usage if they identify daily patterns of behaviour that consume large amounts of energy.

Manipulate the data

The energy supplier provides the half-hourly electricity usage readings in an Excel .csv format file. The data is arranged with the first column having an 8-digit date in ISO8601 format followed by 48 columns for 30-minute intervals each 24-hour day. The file has two years of daily data, so there will be 730 or 731 rows. Manipulation requires each cell to be validated





as holding numeric data from zero value to the daily maximum. The daily maximum could be determined by inspection of a scatter plot of all data or by using Excel functions MAX and MIN to determine the range of data. Other useful functions for indicating the behaviour of the data are: MEDIAN and AVERAGE.

FIGURE 2.62 Scope the range of the data by using Excel functions MAX and MIN.

723	20181224	0.133	0.171	0.164	0.161	0.174	0.151	0.145	0.15	0.127
724	20181225	0.178	0.182	0.18	0.167	0.163	0.158	0.142	0.142	0.13
725	20181226	0.127	0.122	0.134	0.137	0.141	0.147	0.136	0.12	0.12
726	20181227	0.351	0.281	0.208	0.17	0.165	0.16	0.154	0.147	0.152
727	20181228	0.197	0.178	0.163	0.149	0.128	0.123	0.138	0.13	0.122
728	20181229	0.181	0.153	0.131	0.13	0.162	0.124	0.117	0.125	0.121
729	20181230	0.179	0.157	0.272	0.19	0.161	0.161	0.134	0.123	0.138
730	20181231	0.148	0.156	0.178	0.151	0.135	0.13	0.135	0.13	0.123
731	max	1.020	0.976	0.923	1.191	0.716	0.769	0.674	0.639	0.692
732	min	0.085	0.083	0.074	0.074	0.069	0.070	0.073	0.066	0.066
733	average	0.258	0.224	0.201	0.188	0.167	0.156	0.145	0.141	0.138
734	median	0.219	0.192	0.173	0.154	0.145	0.139	0.135	0.132	0.130

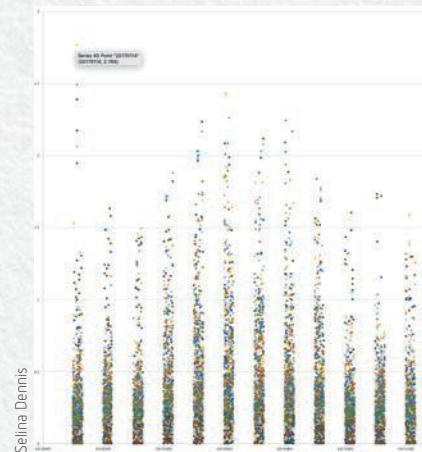
Alternatively, highlight the entire range of data, including dates to get a quick indication of the extent of the data set.

THINK ABOUT APPLIED COMPUTING

2.8

Consider the 2017 energy usage scatter plot in Figure 2.63. When is the greatest energy consumption? There are 12 groups of data points, one for each month. Can you estimate how the energy might have been used? Why are the January figures unusual or unexpected?

FIGURE 2.63 A quick scatter plot of all 2017 data. The highest value can be determined by inspection. The hover Tooltip reveals the greatest 30-minute energy consumption was on 20170114 in interval 40 with 2.765 – in other words, from 7.30 p.m. to 8.00 p.m. on 14 January 2017, when 2.765 kWh energy were used.



THINK ABOUT APPLIED COMPUTING

2.9

Download the 2018 Energy Consumption file from Nelson MindTap, or source your own electricity usage data set.

Determine:

- 1 The greatest 30-minute usage for the year.
- 2 The daily usage and identify when the greatest usage occurred and the value.

Taking a closer look

The default setting for Excel charts will plot each interval as a series, but it is the daily record that is of more interest. The table will need to be transposed for this view (Copy > PasteSpecial > Transpose). The time intervals also need to be added (Edit > Fill > Series ... > Columns > Linear > Begin0.5, StepValue0.5, StopValue24).

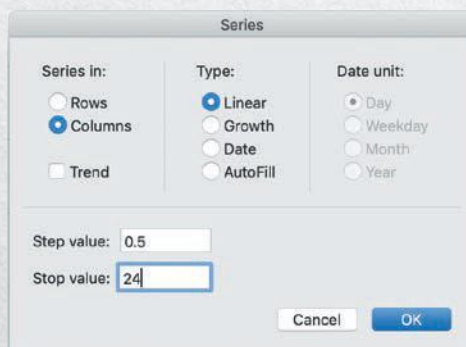


FIGURE 2.64 Insert Interval step values after transposing data.

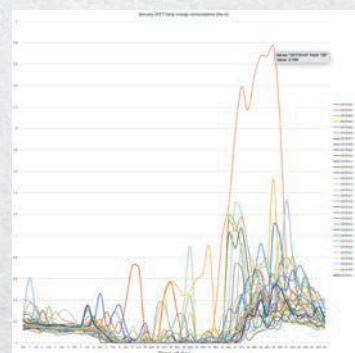


FIGURE 2.65 Energy usage for January 2017 indicates greatest consumption of 2.765 kWh was recorded on 20170114 at 20:00 (8 p.m.) for the preceding 30-minute interval.

Keep it simple

Just because you can graph 17 520 data points, should you? How can the 'story' of the data consumption be explained in a single simple chart? Consumers are probably not so worried about the time of day consumption; rather, the total cost for each month will help them to decide whether they need to change their usage behaviours.

The steps for development of the visualisation will be first to calculate daily totals, then monthly. Bear in mind that the way retailers charge for electricity will depend on time of day, or peak and off-peak time zones. One common peak time zone is 3 p.m. until 9 p.m. inclusive, so after 3 p.m. is the 3.30 p.m. (or 15.5) recording until 9.00 p.m. or 2100 hours.

The data will be required to be processed into peak and off-peak costs. One way to do this is to create a new sheet that refers to the energy used and is multiplied by the corresponding tariff. A quick method of doing this is to create a duplicate worksheet, then to insert the calculated cells in the second sheet with relative references. These can then be copied down and then copied across to calculate daily energy costs. Break down the steps so that you first calculate interval (peak or off-peak), then daily total, then monthly total, and then annual energy cost. The main data of interest will be the 12-monthly totals; however, an interactive chart that allows the user to choose the month to inspect the daily costs might also be worth considering.

Manipulating data with Mathematica

The Monash Tech School Superpowers Program employed Wolfram Mathematica to manipulate the data sets. A function was applied to a 'list of lists' to convert energy usage to costs. Daily totals were then calculated. Interactive controls allow users to choose which consumer profile is shown. Each user has a distinctive colour assigned to assist with identification and consistency.

There is a limit imposed by Excel on the number of series that can be displayed (255). We have 265. We will choose to consider the daily consumption for each month.



Resource
2018 energy
consumption

Excel has a way of quickly summarising a collection of data by using a **pivot table**.

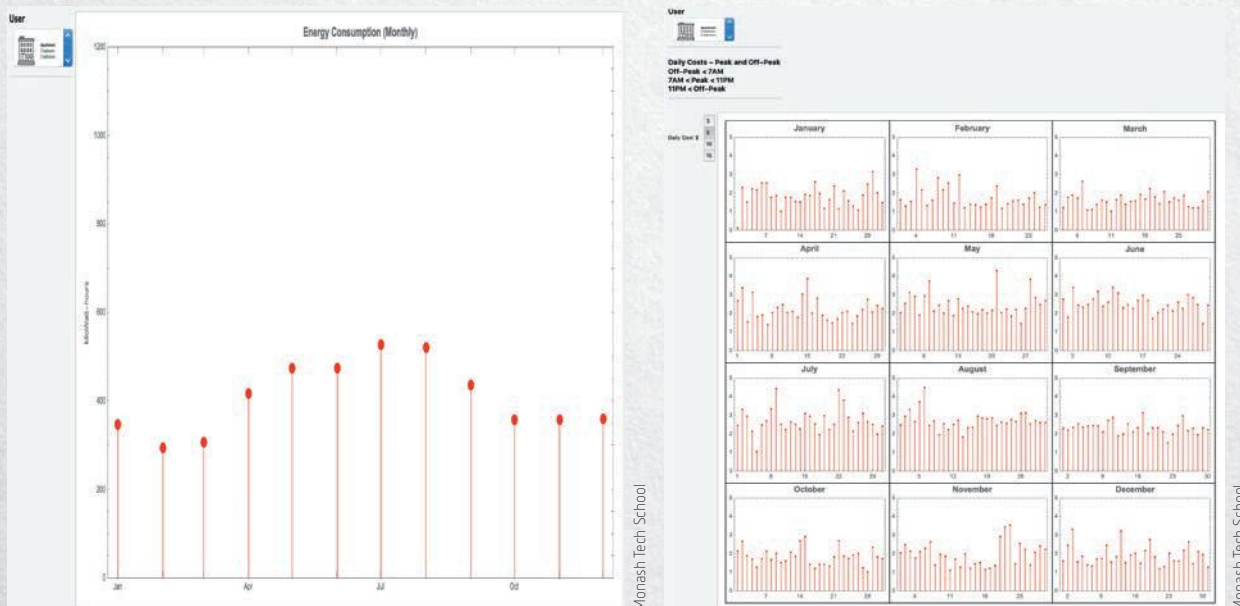


FIGURE 2.66 Year with monthly consumption and Month with daily energy use for the apartment profile (in red)

Further development allowed application of different rates. The 'standing rate' is the default rate that retailers place consumers on if they do not opt into a peak/off-peak arrangement. The difference in costs is shown in a chart and in a table (Figures 2.67 and 2.68).



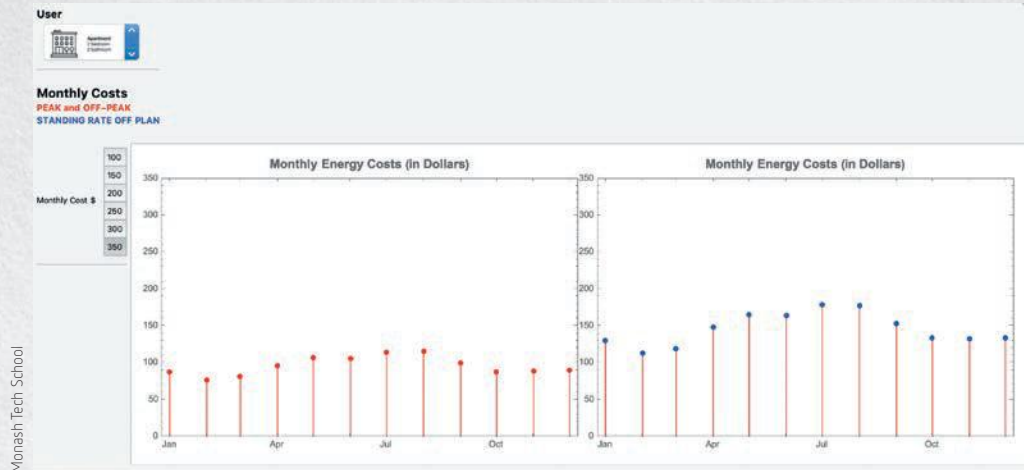


FIGURE 2.67 Comparison between peak/off-peak and the standing rate. Clearly there is a difference in the amount charged for the exact same consumption.

User



Monthly Costs

PEAK and OFF-PEAK **STANDING RATE OFF PLAN**

jan	87.48	jan	130.48
feb	77.07	feb	112.79
mar	81.14	mar	119.56
apr	96.37	apr	148.47
may	107.51	may	165.24
jun	105.69	jun	164.07
jul	114.64	jul	179.32
aug	115.38	aug	177.87
sep	99.36	sep	153.78
oct	87.93	oct	133.69
nov	88.81	nov	132.49
dec	89.97	dec	134.25
	1151.37		1752.04

FIGURE 2.68 A simple table of values, with an annual total, illustrates the differences as clearly as a chart.

Selina Dennis

Interactive Excel charts and dashboards

The interactive features of Tableau have been mentioned previously and are readily apparent in a Tableau workbook. Excel interactivity is not so obvious, and requires some preparation.

The first step is to enable Developer Tools under Ribbon Options or Preferences. (Windows and Mac have slightly different processes.) Once enabled, the Developer tab will appear in the Ribbon above the Formula bar.

Several plots onto the same axes, one at a time

Open the worksheet with the intended data to be charted. Plot all the data onto the same axes, and adjust the 'look' according to previously discussed design principles for charts.



Resource
2017 energy
consumption



FIGURE 2.69 January energy consumption with 31 days on the same axes

The task now is to plot only the selected day or day(s), also known as 'toggling' the chart on/off. We could use a check box for multiple charts or an Option button for a single chart to display this effect.

To achieve this goal, the raw data will need to be duplicated. The check box control cell will be cell A11. The chart will be read from the duplicate, not the raw data, which is never changed to preserve the authenticity and integrity of the data.

In this example, the simpler Option button will be used to show one day at a time.

Go to Developer > Option button.

Click onto the Worksheet to create the Option button. Control-click to open the Formatting dialog box.

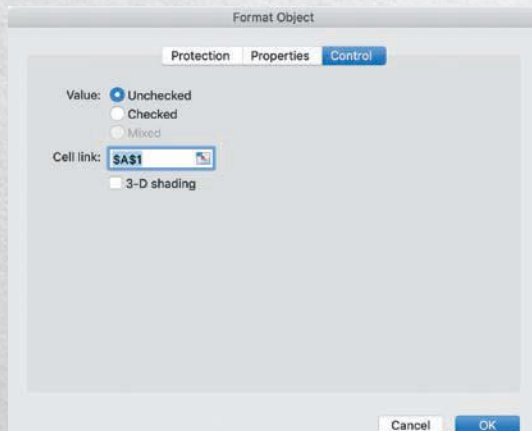


FIGURE 2.70 Set the Cell link to \$A\$1, which will indicate the state of the Option button. Each subsequent Option button will increment the value, up to 31 days in January.





A shortcut to create the Option button is to Control-drag on the previous option button; the value is automatically incremented. Proceed to create 31 Option buttons.

Now for the formula. In the first duplicate data cell, enter the following formula. `IF($A$1=1, enter the corresponding raw data cell – in this case, B2.`

The duplicate cells have been chosen to be 40 cells below the originals for convenience. Then enter comma `NA()`. This will read, if the Option button is 1, or 1/01/2017, then read the first raw data cell, otherwise reads the cell as NULL. In other words, the cell will not plot a point on the chart.

Copy and paste the cell down, and go along cell by cell changing the 'trigger' value to the Option button value for that day. For example, the next cell below will read, `IF($A$1=2,B3,NA())` and the next `IF($A$1=3,B4,NA())` and so on until `IF($A$1=31,B32,NA())`.

Now click, drag and highlight B41:B72, and Copy and Paste across the remainder of the duplicate data. All rows except one will switch to #N/A. Verify the row changes to correspond with the chosen option button.

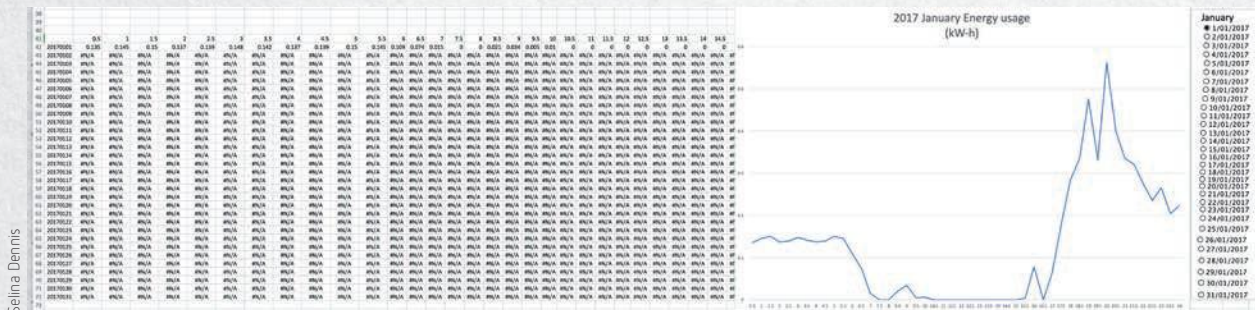


FIGURE 2.71 For display, create an empty text box to place the Option buttons. Control-drag to highlight the Option buttons you want to move.

Why not R and Python?
A similar treatment could be attempted using Python, but the display of the processed data will need to be carefully considered as Python's charting abilities are limited. In contrast, Mathematica has extensive charting functions and comprehensive documentation with worked examples. Although the 'R' language is widely used in industry, it is considered very advanced and beyond the reach of VCE Applied Computing at this time.

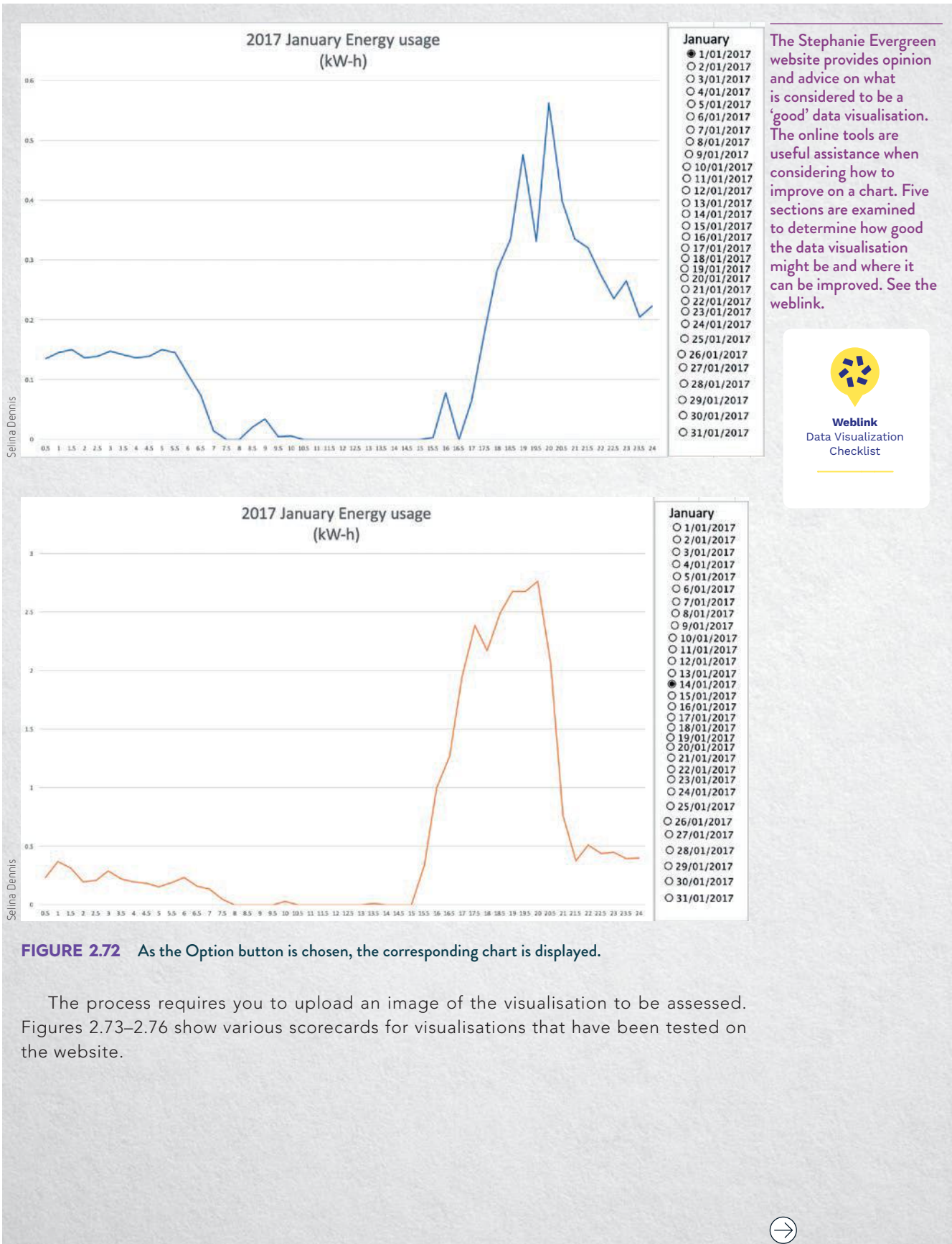
Different charts onto the same worksheet

This is known as a dashboard. A dashboard provides a quick summation of several worksheets. The displays can update dynamically in real time, or some other predetermined time interval. The sources of data can be locally held or streamed. Excel has several streaming functions available under Data > Workbook Connections.

Evaluate your data visualisation chart

There were several design principles discussed earlier in this chapter. Your assessment task for Unit 1, Outcome 1 will be to create an effective data visualisation. So the question becomes, 'How do I know if my dataVis is as effective as it could be?'

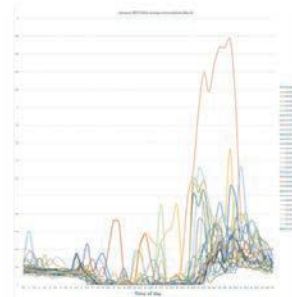
Fortunately, there is an online data visualisation tool (see the margin note and weblink on the next page). The Data Visualization Checklist is a compilation of 24 guidelines for formatting graphs. The checklist is a research-based collaboration by Stephanie Evergreen, Sena Sanjines, Ann Emery and Jennifer Lyons.





Created using <https://datavizchecklist.stephanieevergreen.com>, by Stephanie Evergreen, Sena Sanjines, Ann Emery, and Jennifer Lyons

You're finished! Your score is: 26/48 or, 54.2%.
Great charts score 80-90%.



☒ **Save scores**
We'll use this summary to help validate our checklist.
You'll also get a URL, pointing to this summary to share with your colleagues.
☒ **Make this public**

Fully met

- Q4** Text is horizontal
- Q6** Labels are used sparingly
- Q7** Proportions are accurate
- Q10** Graph is two-dimensional
- Q11** Display is free from decoration
- Q17** Gridlines, if present, are muted
- Q20** Graph has one horizontal and one vertical axis
- Q22** The type of graph is appropriate for data
- Q23** Graph has appropriate level of precision

FIGURE 2.73 The chart had some highly scoring features.

Partially met

- Q5** Data are labeled directly
- Q8** Data are intentionally ordered
- Q9** Axis intervals are equidistant
- Q13** Color is used to highlight key patterns
- Q16** Text sufficiently contrasts background
- Q18** Graph does not have border line
- Q19** Axes do not have unnecessary tick marks or axis lines
- Q24** Individual chart elements work together to reinforce the overarching takeaway message

FIGURE 2.74 The chart had a few 'nearly there' moments and could be easily improved.

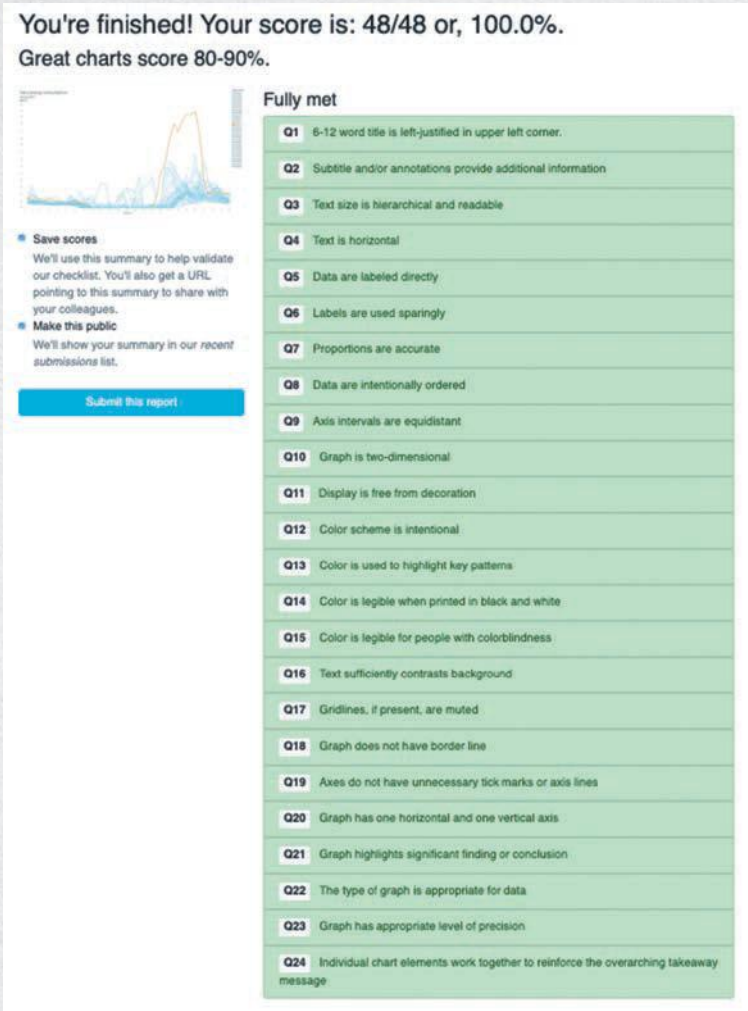
Not met

- Q1** 6-12 word title is left-justified in upper left corner.
- Q2** Subtitle and/or annotations provide additional information
- Q3** Text size is hierarchical and readable
- Q12** Color scheme is intentional
- Q14** Color is legible when printed in black and white
- Q15** Color is legible for people with colorblindness
- Q21** Graph highlights significant finding or conclusion

FIGURE 2.75 There are many areas with room for improvement.

Created using <https://datavizchecklist.stephanieevergreen.com>, by Stephanie Evergreen, Sena Sanjines, Ann Emery, and Jennifer Lyons

The alternative presentation was also assessed.



Created using <https://datavizchecklist.stephanieevergreen.com>, by Stephanie Evergreen, Sena Sanjines, Ann Emery, and Jennifer Lyons

2.10
**THINK ABOUT
APPLIED COMPUTING**

Choose one of your charts, submit a .jpeg image and rate the effectiveness against the 24 criteria. Then attempt to improve your chart to make it a great graph achieving a score of over 80 per cent.

FIGURE 2.76 A very highly rated data visualisation

2

CHAPTER SUMMARY

Essential terms

alignment refers to text: can be left, right, centre or full. Full alignment, which is also known as full justification, refers to text being aligned on the left margin of a column and spaced appropriately so that the last letter in a word on the same line is aligned with the right margin

annotated diagram a visual depiction of how a graphic representation should look; design indicates features such as differences in font size, colour and positioning of objects; the placement of objects must be planned so that a balanced, visually appealing and clear effect is attained

appearance a non-functional principle requirement relating to alignment, repetition, contrast and space

arithmetic mean also known as 'average'; the mid-point of a sample. To calculate: sum of values divided by the number of values. Pseudocode: `quotient[sum[values],count[values]]`

camel case a typographic convention whereby each name begins with a lower-case letter and subsequent words begin with an upper-case letter

chart (also known as a graph) a method of displaying data visually, where the data set is represented as symbols in the chart

constraint a restriction on what can and cannot occur in the creation of a software solution, external to the solution itself

contrast the visual difference in colour or tone between elements on a screen; there should be sufficient contrast between background and text or other page elements to make the information plainly readable

convention standard that has been developed to determine the presentation of documents and other output produced using information systems; rarely static, change over time as changes in technology and business occur; a convention is first decided on and adopted

coxcomb chart (or **rose chart**) a chart that is a variation on a radar and bar chart or pie chart

data set a collection of data with a common factor or purpose

data visualisation the presentation of data in a pictorial or graphical format

date standard a convention that states that all dates should be specified in unambiguous format: `yyyy-mm-dd`

dynamic data visualisation either an interactive with some user control of the view, or one with real-time or regular automatic updates of the data to provide the latest possible data available

economic constraint a constraint on the creation of a software solution that relates either to time or budget, or to both

effectiveness how well a solution meets the needs of its users, measured in terms of completeness, readability, attractiveness, clarity, functionality, accuracy, accessibility, timeliness, report formats, relevance, usability and communication of message

efficiency a measure of how much time, cost and effort have been applied to achieve the intended results

electronic validation machine-validation techniques to ensure the accuracy of data, which can be built into software, such as spell checkers. Types of electronic validation include range checking, existence checking and data type checking

filtering inclusion of certain data in, or its exclusion from, a data set. Logic functions can be applied in choosing matching data that satisfies the conditions

Form Controls Excel developer tools that change the settings chosen by the sheet or chart

format factors that govern the appearance of a digital solution, such as the typeface and background colour

functional requirements requirements of a digital solution that are directly related to what the solution will do

Hungarian case a typographic convention that uses the object type to begin the variable name. Its use is no longer recommended

IPO chart (input–process–output chart) a defining diagram that shows how data is processed into meaningful information; it identifies what data is required for the solution (input), what information the solution needs to produce (output), and the processing steps required to transform the data into information, or the function of the solution

macro script that can automate certain actions; often used in applications to perform repetitive, predictable actions

manual validation the process of checking data entered for accuracy when done by a person, rather than by a machine

median middle value when the number of values is odd; when the number of values is even, it is the average of the two middle values

mode most frequent value

non-functional requirements characteristics that the user or client would like the digital solution to have; often tied to solution constraints

non-technical constraints factors that may limit or restrict solution development, specifically related to legal, social, usability or economic factors

outlier a data point that is either much bigger or much smaller than other data points in a set of data (e.g. on a graph)

Pascal case a typographic convention that is similar to camel case, with an upper-case letter at the start

pie chart circular chart that represents 100 per cent of a whole. Segments represent the proportion of the whole.

pivot table a feature of spreadsheets that allows quick summarising of data

qualitative data categorical or non-numerical data that has been counted rather than measured; collected through interviews, focus groups or observation

query a question that interrogates the database to find records

rose chart (or **coxcomb chart**) a chart that is a variation on a radar and bar chart or pie chart

Sankey diagram a style of diagram used to illustrate flows of quantities by widths of arrows, typically energy flows or production losses

Spreadsheet Standards Review Board (SSRB) a body that recommends best-practice methods for spreadsheet model development

SQL Structured Query Language; a commonly used relational database

standard deviation (SD) normal distributions measure the separation distance or offset or deviation by calculating a standard deviation or sigma (Greek letter 's'); six sigma includes 99.9 per cent of a normal population

static data visualisation a visualisation that has a fixed view with fixed data; it does not change

sum the total of all the values (i.e. values added together or totalled)

technical constraints constraints related to the hardware and software available for a project

usability a functional requirement principle relating to user capabilities and the ease of use of the solution; all spreadsheet solutions, including graphic representations of numerical data, need to be user-friendly; the user should be able to access the required information with ease and efficiency

validation the process of inspecting data, while it is being input into a solution, to check if the data is reasonable

Important facts

- 1 While a spreadsheet has just a table view, a database can show a table view or a record view that shows each field value for a record.
- 2 In Excel, a name can be given to a cell or range of cells.
- 3 Many of the functions of a spreadsheet can achieve similar results to a database, but they are not equivalent.
- 4 Spreadsheets require the data set to be held in the RAM of the local computer.
- 5 Database applications have no limit to the size of the files that can be accessed.
- 6 Databases are organised collections of structured information or data.
- 7 Tables in a database are similar to worksheets in a spreadsheet. Columns in spreadsheets and databases are comparable.
- 8 Primary keys are keys that can uniquely identify each record in a table. Foreign keys are used to establish relationships between tables, like a reference.
- 9 Both spreadsheets and databases have in-built functions that can quickly calculate summary information based on selected data.
- 10 SQL is widely used to help sort and filter data and information. It is designed to communicate with a database using programmed commands, called queries.
- 11 Query design tools include joins such as inner join and outer join. These allow for tables to be combined based on related columns.
- 12 Data validation is required to increase the accuracy and reasonableness of the data within a solution. Two types of data validation are manual and electronic.
- 13 Data visualisations make data patterns and relationships clearer than lists of numbers.
- 14 Dynamic interactive data visualisation manually or automatically updates data and has user interaction.
- 15 Choose types of charts with care so they visualise data clearly and accurately.
- 16 Design principles affect the functionality and appearance of solutions.
- 17 There are functional and non-functional design principles.
- 18 A constraint is any factor that limits the data visualisation solution.
- 19 Design tools are used to represent the functionality and appearance of your solution. These include annotated diagrams and mock-ups, IPO charts and query designs.
- 20 Annotated diagrams such as wire frames provide a visual depiction of how your visualisation solution should look. Wire-frame details are usually in black and white and show location, content and function.
- 21 A mock-up provides format information, including colour, size, font and 'look and feel' appearance.
- 22 An IPO chart can be used to show how data is processed into meaningful information.
- 23 An IPO chart identifies what data is required for the solution (input), what information the solution needs to produce (output), and the processing/manipulation activities required to transform the data into information.
- 24 Query designs involve selection, joining, aggregation, insertion, updating and deletion.
- 25 Conventions are general rules that are followed when using a particular format.
- 26 Common conventions include clear title, axis labelled, key or legend used, name of the author and the source of the data, unit of measurements shown and matching colours.



Spreadsheets, databases and SQL

- 1 Why would a developer use named cells within a spreadsheet solution?
- 2 What benefit does a database solution have over a spreadsheet solution?
- 3 What is the difference between a primary key and a foreign key?
- 4 Write a query to select all columns from a table named 'students'.
- 5 Write a query to select all students with the 'year' value of 'Year 11' from a 'students' table.
- 6 Write a query to join 'students' and 'classes' tables based on a shared 'class_id' column.
- 7 Write a query to update the 'year' column in a 'student' table to 'Year 11', where the 'student_id' is 17562.
- 8 Write a query to change all classes with the 'class_name' of 'History' to 'World History' in a 'classes' table.

Types and purposes of data visualisation

- 9 Define 'data visualisation'.
- 10 List three types of data visualisations.

Design principles

- 11 What are the two main categories of design principles for data visualisation?
- 12 Explain the difference between functional requirements and non-functional requirements.
- 13 Define 'functional constraint' and 'non-functional constraint'.
- 14 Outline the purpose of a design tool.
- 15 Explain the difference between appearance design tools and functionality design tools.
- 16 List two design tools for appearance.
- 17 Discuss the advantage of creating a number of annotated diagrams representing the data in a different format.
- 18 Explain the purpose of an IPO chart.
- 19 Outline a situation in which using a mock-up over an annotated diagram would be appropriate.

Formats and conventions

- 20 Define 'format'.
- 21 Define 'convention'.
- 22 Explain the difference between formats and conventions.
- 23 List four conventions usually followed in data visualisations.



APPLY YOUR KNOWLEDGE



Resource
Energy sample file

- 1 There are two main categories for data visualisations: explanation and exploration. What is the difference from the viewer point of view?
- 2 State which type of chart you would choose if the data were:
 - a {temperature and time}
 - b {stocktake data, numbers of each product}
 - c {temperature recorded every five minutes, rainfall every half hour}
 - d {contribution share of hours worked in a week}
- 3 Download the *EnergySampleFile* in .xlsx or .csv format from Nelson MindTap. Use spreadsheet or database software to do the following.
 - a Create a monthly report on energy usage.
 - b Estimate the cost of electricity for the chosen month where the rate for peak is 40.2 cents per kWh and off-peak is 17.5 cents per kWh. In addition to the usage charge, there is a service charge of \$1.02 per day. Peak is the period between 3 p.m. and 9 p.m. each weekday. Weekends are charged at the off-peak tariff.
 - c List some of the assumptions you have made in calculating the monthly account.
 - d Create an IPO chart for the spreadsheet calculations.
 - e Create a monthly chart showing the daily energy usage and a chart of the daily energy cost using a bar chart and a line chart with two vertical axes.
 - f Comment on some observations about similarities and differences of the two entity values (energy usage and cost).

PREPARING FOR Unit 1

OUTCOME 1

Your teacher will provide you with solution requirements and designs for which you need to organise appropriate data, analyse it and present the findings as data visualisations.

As part of the solution development you will validate your data and apply appropriate formats and conventions to the data visualisations.

Assessment

All assessments at Units 1 and 2 are school-based. Procedures for assessment of levels of achievement in Units 1 and 2 are a matter for school decision.

For Unit 1 students are required to demonstrate two outcomes. As a set these outcomes encompass the areas of study in the unit.

Suitable tasks for assessment in this unit may be selected from the following:

Unit 1, Outcome 1

- a folio of exercises to demonstrate the learning of database, spreadsheet and data visualisation software tools
- a solution including the use of database, spreadsheet and/or data visualisation software tools in response to teacher-provided solution requirements and designs
- a personal portfolio to showcase the development of databases, spreadsheets and data visualisations.

Victorian Curriculum and Assessment Authority, Victorian Certificate of Education Applied Computing Study Design 2025–2028, VCAA, 2024, p. 29.

SAC Create a data story

A possible culmination task for school-assessed coursework would be the creation of a data story using data visualisation software. Students may choose from a range of teacher-provided data sets. Alternatively, students may identify a data set and cite full details of the source.

The process of validating data, manipulating the data set and presenting a description of the data behaviour would be the focus of the task. Students will document the development of the data visualisation from initial analysis through to an evaluation of the effectiveness of the final visualisation. A simple journal or blog space would be sufficient to record thoughts, steps and decisions.

Students would be expected to use suitable software (for example, Excel and Tableau) to first validate, and then manipulate, the data set before the creation of appropriate charts to convey a story about the data. While the data set will be static, the presentation may provide users with choices to select views of the data. This may be achieved through interactive controls or by having several charts accessed by a menu.

Clarity of design and ease of operation will be key performance criteria.

The format for submission will be discussed with your teacher. A range of possible formats are feasible, and local considerations will determine the final requirements.

Your teacher will also provide assessment criteria, a range of appropriate data sets or a source of data sets, journal or blog space details and a timeline for completion of the Unit 1, Area of Study 1, Data visualisation SAC.

Designing software

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

- KK1.2.1: emerging trends in programming languages and artificial intelligence-based (AI) code generators for the development of software solutions
- KK1.2.2: characteristics of data types
- KK1.2.3: types of data structures
- KK1.2.4: characteristics of functional and non-functional requirements, constraints and scope
- KK1.2.5: design tools for representing the functionality and appearance of solution designs
- KK1.2.6: key legal requirements relating to intellectual property and copyright while designing and developing software.

Victorian Curriculum and Assessment Authority, Victorian Certificate of Education Applied Computing Study Design 2025–2028, VCAA, 2024, pp. 27–8

FOR THE STUDENT

This chapter relates to VCE Applied Computing Unit 1, Area of Study 2: Programming. It introduces basic programming concepts such as software development tools, functions and capabilities of software and hardware, design tools, types of programming languages, and universal programming ideas such as pseudocode, data types and data structures. You will also begin learning a specific programming language that you will use to develop a response to a design brief for Unit 1, Outcome 2. This will involve responding to a design brief in relation to functional and non-functional requirements, constraints and scope. You will also need to design and apply suitable testing and debugging techniques as well as document the functioning of the software solution by the use of internal documentation.

This chapter introduces you to some of the universal theoretical concepts behind programming that are required for Unit 1, Outcome 2. It does not assume knowledge of, or use source code from, any actual language. Unit 1, Outcome 2 requires that you apply all stages of the problem-solving methodology to design, develop and evaluate a software solution in response to a design brief provided by your teacher. This design brief should include functional and non-functional requirements, constraints and factors that determine scope. It will allow you to draw on knowledge of basic programming concepts, such as storage, logic, loops and calculations. Programming concepts such as logic, loops and calculations are covered in detail in Chapter 4.



Programming and scripting languages

Programming languages are used to give instructions to computer processors so they can calculate useful information or carry out tasks for humans. Whether your phone is playing an MP3, your car is turning on its anti-skid braking, or McDonald's is calculating staff wages, programming languages are needed.

Scripting languages conveniently store sequences of instructions that, alternatively, could be entered one at a time. Like human languages, there are many programming languages, each with distinctive grammar, punctuation and vocabulary. Most programming languages have special abilities or strengths that make them more useful than other languages for a particular task.

Professional programmers typically know a handful of languages and choose the best one for each job based on its strengths and weaknesses. Choosing the languages to learn is a big decision, but remember that learning one language makes it easier to learn others. The most popular programming languages include C (C++ or C#), Python, Java, JavaScript, Perl and PHP, SQL, Visual Basic and Swift.

- C, C++ or C# is used for writing low-level utilities and fast applications. This could include operating systems, embedded microcontroller programs, web-based applications and games.
- Python is an interpreted object-oriented programming language used for web and app development.
- Java is used for web applications and web services and for building Android apps.
- JavaScript is a client-side scripting language for websites.
- Perl and PHP are used in website and network programming.
- SQL, or Structured Query Language, is a scripting language for database programming.
- Visual Basic and Visual Basic.NET are used widely to create applications for Windows-based computers.
- Swift is a programming language for macOS, iOS, watchOS and tvOS.

While programming languages may differ, they all do basically the same job: they control a digital system such as a computer, tablet or smart device.

Programming languages differ in the amount of direct control they give over a computer's hardware and operating system. With a **high-level programming language** such as Visual Basic or Python, programmers avoid having to worry about complex details of the structure of actual disk files or where data is stored in memory. High-level languages are simpler to use, but lack the control of complex, but more difficult to learn, low-level languages. Conversely, a **low-level programming language** such as C or machine code requires more skill and knowledge from the programmer, but allows more direct control of the workings of a computer.

High- and low-level programming languages each have their uses. To write a simple alarm clock program, a high-level language is fine. To write a device driver to control a printer, only a low-level language will do.

3.1 THINK ABOUT APPLIED COMPUTING

Research three popular languages to discover their origins. What did existing languages lack that led to the need for the new languages?

Software development tools

To develop software, you need a number of basic, essential tools, including an **editor**, a **compiler** or **interpreter**, and a **debugger**. The following section discusses these in more detail.

An editor is a specialised application software that is used for creating human-readable **source code** or, rather, human-readable programming instructions. Code editors come with specialist features designed to make programming easier by, for example, highlighting programming keywords, using code completion to reduce typos and other mistakes, detecting unbalanced parentheses and adding line numbering. Programming editors, such as the one shown in Figure 3.1, show colour-coding, indenting, collapsible text and line numbering.

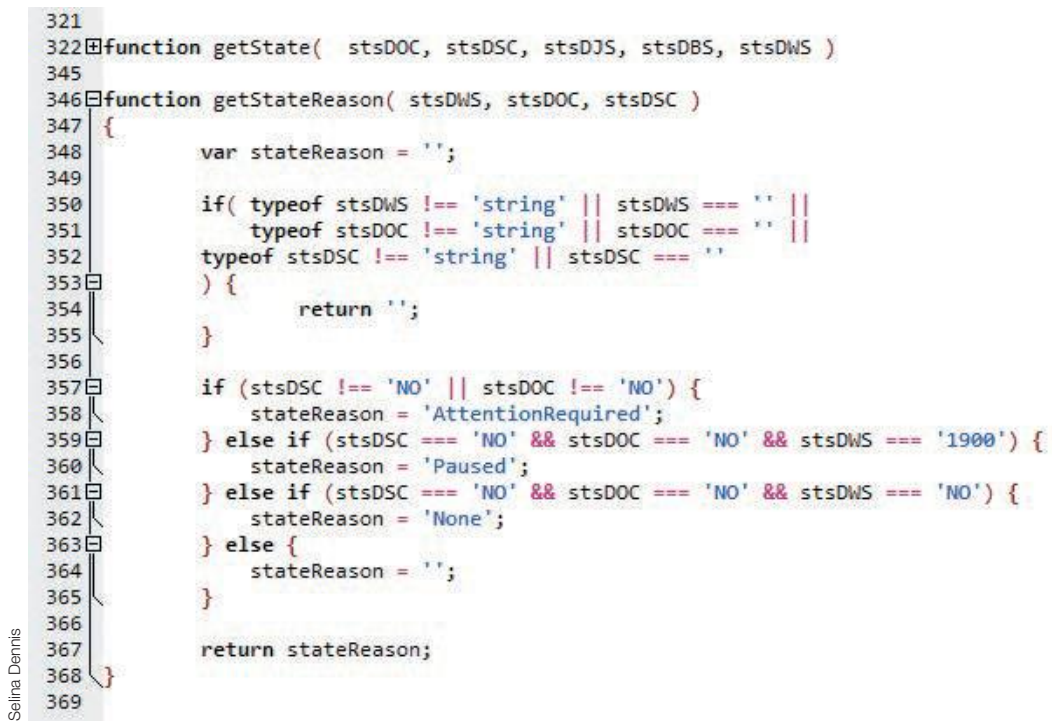


FIGURE 3.1 A programming editor

A compiler converts source code into executable programs that a computer can carry out – that is, that a particular CPU and operating system, such as Windows or macOS, can understand. An interpreter directly executes source code without needing to compile it beforehand.

Executable code compiled for one **platform** will typically not work on another without being ported (re-compiled for another platform). Porting can be hard and expensive work; this is one reason why many apps are available for Windows but not Mac or vice versa.

A debugger helps programmers to find ‘bugs’, or programming errors. Sometimes debugging can take as long as the original programming time, or even longer if the program has been poorly designed. Debuggers may:

- highlight incorrect **syntax** (programming expression) and show how statements should be expressed

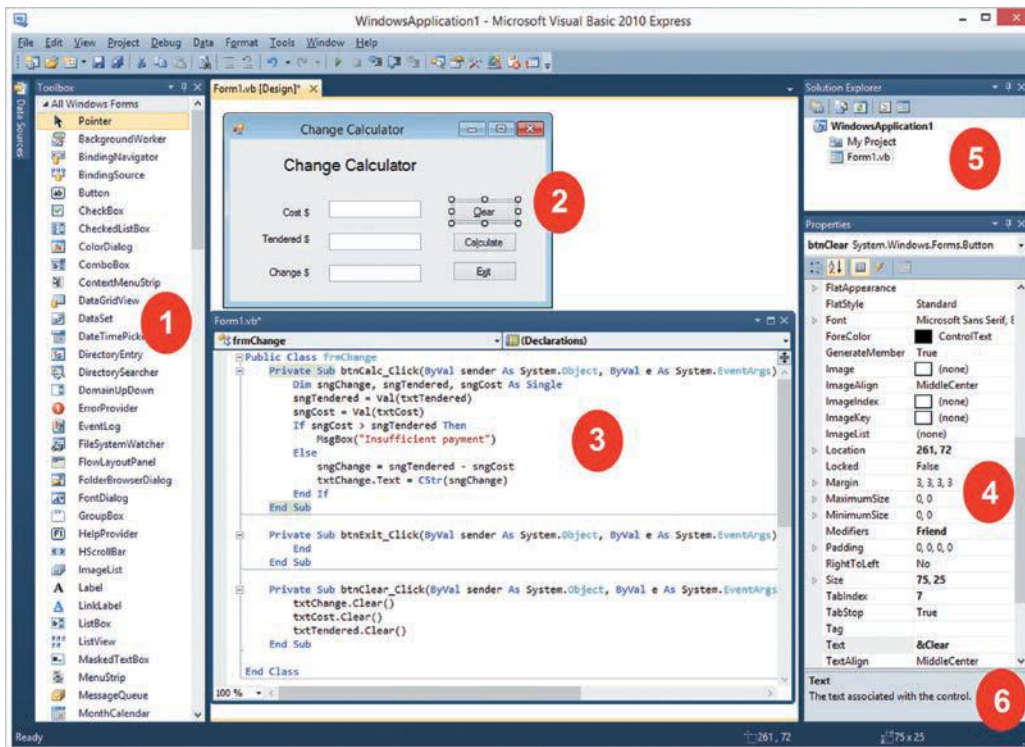
Part of the difficulty of porting is that programs must often be substantially changed to work under a different operating system.

- allow programmers to set breakpoints in code, where the compiler will stop and let the programmer inspect the current values of variables
- allow line-by-line stepping through code so developers can find exactly where a problem arises.

Few programmers today use separate editors, compilers and debuggers. Most use an **integrated development environment (IDE)**, such as Microsoft's Visual Studio Code (VS Code), that combines the development tools into a single package. VS Code accommodates several popular languages.

Figure 3.2 shows:

- 1 the toolbox of **graphical user interface (GUI)** objects the programmer can insert into the program
- 2 the form, which will be the visible **interface** for the program's user
- 3 the code window, which the programmer uses to instruct the program how to act when an event takes place, such as the clicking of a button
- 4 the **properties** of the selected **object**, which let the programmer modify an object's characteristics or behaviour
- 5 the project manager, with which the programmer can manage the various files and components related to the program
- 6 onscreen help to give brief reminders of what the currently selected object is like.



Selina Dennis

FIGURE 3.2 Example of an IDE

Emerging trends in programming and AI

Emerging trends in programming languages and AI-based code generators have already begun to transform how we create software.

Low-code development

Low-code development platforms allow developers to create applications with minimal hand-coding by using visual interfaces and drag-and-drop tools. This approach streamlines the development process, making it more accessible for those with less traditional programming experience while also speeding up the creation of software solutions. By reducing the amount of manual coding required, low-code platforms enable quicker iterations and deployments, allowing developers to focus more on the logic and functionality of applications rather than the intricacies of coding.

For example, Microsoft Power Apps enables users to build apps quickly through a simple interface where they can drag and drop elements, such as buttons and forms, without writing extensive code. This approach allows even non-programmers to develop functional applications, which can be especially beneficial in business settings where quick, custom solutions are needed to address specific problems.



Weblink
Microsoft Power Apps

FIGURE 3.3 Some of the options in Microsoft Power Apps

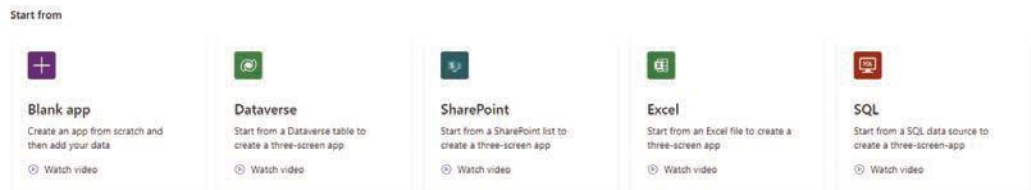
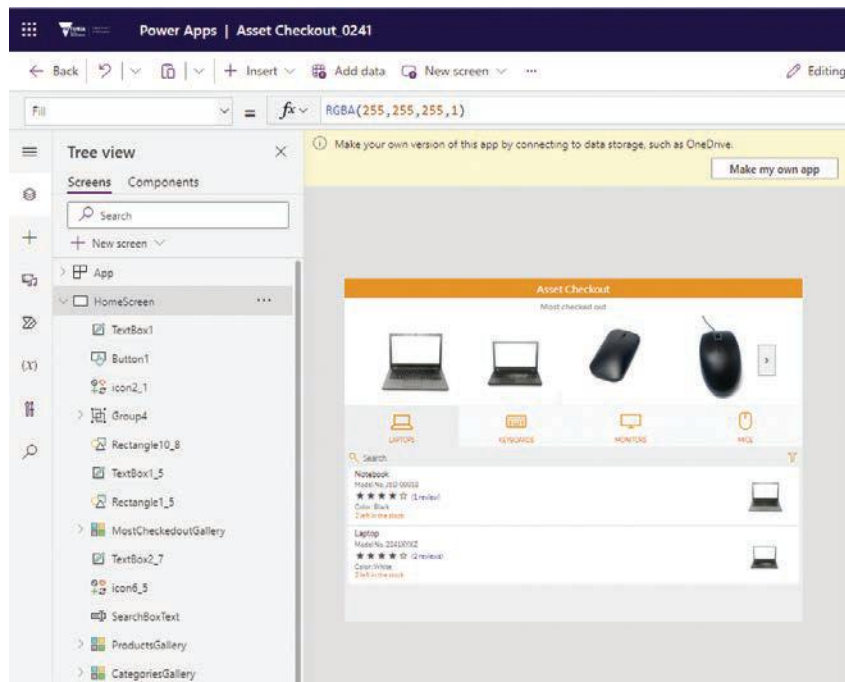


FIGURE 3.4 A Power Apps template for a shopping checkout



Readability and maintainability improvements

AI-based code generators enhance the efficiency and quality of code by automatically generating code snippets or entire functions based on given requirements or contexts. These tools often leverage natural language processing to interpret human instructions, making it easier to write, understand and maintain code. Improvements in readability and maintainability are also at the forefront, with modern programming languages and tools focusing on cleaner, more understandable syntax and better documentation practices. This not only makes the code more accessible to current developers but also ensures that future developers can easily comprehend and modify it, encouraging a more sustainable and collaborative development environment.

For example, GitHub Copilot, an AI-powered code completion tool, can suggest entire lines or blocks of code as a developer types, based on the context of the project. This not only speeds up coding but also helps maintain consistent coding styles and standards.

View a demonstration of GitHub Copilot by navigating to the GitHub website, clicking on the Product tab, All features then Copilot.



Weblink
GitHub Copilot

Characteristics of a software solution

For a problem to be satisfied by a software solution, certain characteristics must be provided for:

- scope
- functional requirements
- non-functional requirements.

A solution will be considered acceptable if the characteristics have been identified prior to any writing of program language code, and addressed with the output of the program achieving its intended purpose.

Determining scope

The **scope** outlines the boundaries or parameters of the solution so all stakeholders are aware of exactly what the solution will contain. The scope of the solution consists of two elements: what the solution will do and what the solution will not do.

What the solution will do

What the solution will do is set out in a list of all the solution requirements (both functional and non-functional) that will be included in the solution.

Many house and land package contracts state exactly what is included with the package (in scope) and exactly what is not included in the package (out of scope). For example, kitchen floor tiles might be included, but the garden might not be landscaped.

3.2 THINK ABOUT APPLIED COMPUTING

What problems do you think a clear scope of solution could avoid later in the project?

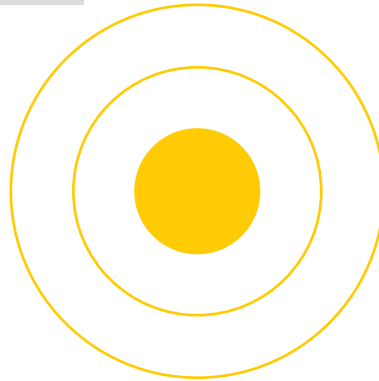
What the solution will not do

What the solution will not do is set out in a list of all the solution requirements that will not be included in the solution.

Usually these are solution requirements that the client initially sought but that, because of constraints, have been left out of the solution project.

At the start of the project, it helps to outline what will and will not be included in the solution to prevent arguments later in the project between the client and the developer.

Scope target



The area close to the centre is relevant to the scope of the project; further away outside the inner circle is not.

FIGURE 3.5 Scope of solution

Scope creep happens when a client changes the scope of a project (by increasing it) during the life cycle of the project. This can be very expensive to a developer in terms of time and money, as it means the project will take longer to complete, often without any extra money being provided by the client.

A statement of scope for a solution might be as follows: ‘The solution will display population data of towns in Victoria in a visual format and graphically represent the distances between those towns. It will be created to be user-friendly and, for privacy reasons, it will not display any personal details.’

A statement for something that is out of scope might be: ‘The requirement that allows the user to zoom in on a particular region or town will not be included in the project because of economic factors, but may be added at a later stage.’

In a design brief, aspects of the solution that are within scope are described without technical detail. There may be general descriptions of what the solution will contain in relation to functional and non-functional requirements. Determining what may be out of scope – for example, deciding what functionality may be delayed for later releases due to time constraints – normally occurs during the analysis stage.

Functional and non-functional requirements

Solution requirements are what the client needs from a solution – that is, what the system must do. These can be broken down into **functional requirements** and **non-functional requirements**.

Functional requirements

Functional requirements are directly related to what the solution will do. They typically involve: calculations; data processing; opening, reading and writing to files; data manipulation such as image editing; and other specific functionality required within the system.

Some examples of functional requirements for software solutions are that they:

- save customer data to a file
- calculate discount values on products
- set an alarm to go off at a particular time
- load a set of jobs into a timetable.

Functional requirements are usually described in terms of the inputs required, the sequence of operations that will be performed within the function, and the output(s) of the function after processing has occurred. These written descriptions often include dot-point descriptions for the purposes of clarity. An example of a functional requirement for a music performance system can be seen in Figure 3.6.

1. Add Performance

This screen will allow the entering of performance information. Information consists of the performance name, date and theme, as well as a unique performance ID (numeric). Also recorded is whether the performance is a concert, and, if so, the theme of that concert.

Inputs:

- Performance ID (unique)
- Performance Name, Performance Date
- Whether it is a concert or not
- Concert Theme

Sequence of Operations:

- Input will come in from the User Interface, all items except concert theme are required fields
- Themes should be retrieved from a file containing all available themes so a user can select one
- No two concerts should have the same ID
- A theme only needs to be selected if it is a concert
- No concert should be added if a concert with the same theme has occurred in the last six years
- Once data is entered and validated, it should be saved to a performance file

Outputs:

A pop-up box should be displayed showing the success or failure of saving the concert information. Once information is saved, the user should be returned to the main menu.

Selina Dennis

FIGURE 3.6 Functional requirement for a music performance system

Non-functional requirements

Non-functional requirements are those characteristics that the client would like the solution to have, but that do not affect what it does. Non-functional requirements are often tied to the constraints of the system. They can be categorised in terms of usability, reliability, portability, robustness and maintainability.

Non-functional requirements must be measurable – this means that it must be possible to test them to see if the requirement is met.

Determining non-functional requirements usually involves discussions with a client, such as asking if the software must work on different operating systems, or asking who the users of the system will be and the level of technical experience they have.

Usability

Usability relates to how easy a system is to learn and use. This is typically described in terms of efficiency and effectiveness. Common factors of usability include the clarity of the user interface and the intuitiveness of the functions within the system. The success of a system's usability is often measured in terms of user satisfaction. An example of a non-functional requirement related to usability is that 'users should be able to use all basic functions after one hour of training'. This non-functional requirement is highly reliant on the skills, expertise and needs of the intended users of the system.

Reliability

The **reliability** of a software solution relates to how much it can be depended upon to function as designed, and for how long. Typically, this requires that the software is deemed **fit for purpose** over time and that it is resistant to failure.

The reliability of a system is generally expressed as a probability measure, where $\text{reliability} = 1 - \text{the probability of failure}$. An example of a reliability measure is the prediction of the **uptime** of a system or solution, such as 'the system should have a 99.9 per cent uptime over any 12-month period'; another might be 'the ability to add a new customer should be available to staff members during working hours on weekdays'.

Portability

The **portability** of a software solution relates to how easily it can be used in different operating environments. This can involve measuring how easy it is to move from one system to another system with the same architecture; how easy it is to reinstall a program on a new system; and the ability to use the same software on multiple operating systems and platforms.

The most time-consuming element of portability is the requirement to write software that will work on multiple operating systems and platforms. This typically means that a developer must separate the user interface from the core functionality and logic so that it is easy to create a new user interface for a new system – for example, creating an application that works on a mobile device as well as on a desktop computer.

Portability is increasingly important as mobile computing becomes the norm; users often expect that applications will work on mobile technology as well as on desktop technology.

Robustness

The **robustness** of a system relates to how well a software solution responds to errors that occur when the software is being used; that is, it should perform correctly in every situation a user encounters. Robustness is therefore an evaluation of the error-handling techniques built within a software solution. For example, the robustness of a piece of software would be evaluated by assessing how it responds to bad input from a user. Robustness is closely linked to the use of validation techniques as these can help prevent errors from occurring when users enter unexpected or invalid input. An example of a non-functional requirement related to robustness is that 'the system should reject invalid data entered by a user'.

Robustness is measured in terms of the number of failures, crashes and errors that occur while a system is running.

Maintainability

Maintainability is related to how easy it is to look after software once it is in use. This can involve fixing errors in the code, maximising efficiency and reliability, installing the software on new systems and, in some cases, expanding on the current functionality with new functionality. Simply put, maintainability is measured in terms of how easy it is to fix, modify or change the software once people are using it.

Often, maintainability is measured in terms of the number of hours a developer or administrator spends on the system to keep it running after it has been put in place. An example of a non-functional requirement related to maintainability is that ‘fewer than 10 hours should be spent per quarter on maintaining the system’.

Constraints

Solution **constraints** are factors that may limit or restrict solution requirements, determined during analysis. At the stage when a design brief is created, these constraints are described only in general terms. Typically, constraints involve economic, technical, social, legal and usability factors.

Economic constraints

Economic constraints include time and cost.

The deadline by which the user or client needs to have the solution operational will define the time available to design and develop the solution. The longer this is, the more time there is to complete an in-depth analysis and detailed designs and to develop advanced features of the solution. The shorter the time frame, the more quickly each stage in the problem-solving methodology needs to be completed.

Meanwhile, the budget (money) available to complete the project may affect the hardware and software (digital systems) available for use, the number and range of staff who are available to work on the solution and even the data used as input (if the data sets required need to be purchased).

A lack of either time or money may result in a re-evaluation of the user’s requirements, or a re-evaluation of how the requirements can be achieved.

Technical constraints

Technical constraints are constraints related to the hardware and software available for the project. Available hardware and software, memory and storage capacity, processing and transmission speeds and security concerns are all examples of possible technical constraints.

For example, developers need to keep in mind that smartphone users may not always have access to a high-speed network connection, so they need to ensure that any animated data visualisation solution does not require a large amount of bandwidth to download and view.

Non-technical constraints

Non-technical constraints relate to areas other than hardware and software: social constraints and usability constraints. Usability and the user’s level of expertise (which is a social factor) are examples. If a solution is being developed for users with little digital systems expertise, this may restrict some of the requirements that would involve complex manoeuvres to complete. Creating a solution for a child audience may restrict the method used to input data into the solution.

3.3 THINK ABOUT APPLIED COMPUTING

List three other technical constraints that developers of smartphone apps need to consider when developing a product.

Privacy laws were discussed in Chapter 1; copyright laws are further discussed in Chapter 6.

Legal requirements are another type of non-technical constraint. Privacy laws may restrict features linked to displaying personal data in the solution, or to collecting data from the devices of someone using your solution. Copyright laws may restrict features that allow other users to upload content to the solution.

Legal constraints

Laws are passed by governments at the federal level and at the state level. Each parliament debates and enacts legislation, which is then enforced by the law courts for each jurisdiction. Federal (or Commonwealth) laws override state laws. In the absence of federal laws, the state law will apply.

In Victoria, copyright laws are applied under federal laws. The *Copyright Act 1968* (Cwlth) has been amended several times to keep up to date with changes in technologies. The latest amendment was in 2014, to include digital music and computer programs.

When creating software or a digital product, legal requirements and **intellectual property rights** must be considered to avoid prosecution under the *Copyright Act 1968*. Software is considered a literary work and is covered by the Copyright Act. Copyright is automatically applied at creation and does not need to be registered.

Definitions under the Copyright Act include:

Section 10

computer program means a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result.

literary work includes:

- (a) a table, or compilation, expressed in words, figures or symbols; and
- (b) a computer program or compilation of computer programs.

Section 47AB

computer program includes any literary work that is:

- (a) incorporated in, or associated with, a computer program; and
- (b) essential to the effective operation of a function of that computer program

Section 10 and Section 47AB of the Copyright Act 1968 (Cth). Source: <https://www.legislation.gov.au/C1968A00063/2019-01-01/text>

The holder of copyright depends on the circumstances. Usually, the author of the computer code is the owner, unless there is a contract of employment (section 35) where the ownership is vested in the business. If code is produced that uses code libraries owned by others, the ownership may be in dispute. Section 10 covers joint ownership. Where code is produced by a team effort, the ownership is shared.

If software is written that uses a code library – for example, where the IP (intellectual property) is owned by someone else – permission must be acquired to avoid ‘breaking’ the copyright. Sometimes permission costs nothing; however, many software companies charge for permission or the ‘right to copy’. These permissions are either a single fee for that edition or a yearly subscription.

One such use often extends to the programming languages. Python has terms and conditions that permit the use of the programming language at no cost. Other programming languages may have an annual fee for commercial products. The programming language Java has a free hobbyist licence; however, any commercial application has a yearly licence fee.

Always check the terms and conditions of any third-party software prior to including it in your project. Sometimes ‘look and feel’ can also be copyrighted. When a program is rewritten in a different computer language with identical instructions to achieve a similar outcome, it is no longer covered by the original copyright. Often reverse engineering is the method adopted to avoid copyright restrictions. This is sometimes referred to in the terms and conditions and expressly prohibited; however, the *Copyright Act 1968* (section 47B-3a) permits the study of software for the purposes of analysis.

Characteristics of data types

In programming, classifying a variable by **data type** is a way of determining the data that variable can contain, as well as how that variable can be manipulated – that is, what it can do, and what can be done to it. While programming languages vary greatly from each other, data types do not; they are consistent across all programming languages. When programming, it is important not only to choose an appropriate data type when creating a variable, but also to select the most efficient data type. For example, it is not efficient to select a numeric data type that supports decimal places when creating a variable if that variable will only ever contain whole numbers. Similarly, storing a number as a string is not as efficient as storing it as a numeric data type, even if there exists a method of converting strings to numbers.

Numeric

The **numeric** data type consists of whole numbers, referred to as **integers**, and decimal numbers, referred to as **floating point numbers**. Integers can be referred to as *unsigned*, which means they can only store positive whole numbers, or *signed*, which means they can store both positive and negative whole numbers.

All numeric data types can have mathematical operations performed on them. The fundamental operations shown in Table 3.1 are the most common ones.

TABLE 3.1 Fundamental data type operations

Addition	+
Subtraction	−
Multiplication	*
Division	/
Whole number division (quotient)	//
Remainder after division (modulo)	%
Powers	**
Assign values	= or ←

When more than one operation appears within a line of code, the order of operations follows the same rules as in mathematics: brackets, orders, division and multiplication, addition and subtraction – or BODMAS. If two operators have the same precedence, they are evaluated from left to right.

Numeric data can also undergo comparisons, with those shown in Table 3.2 being the most common.

TABLE 3.2 Data type comparison operations

Less than	<
Less than or equal to	<= or ≤
Greater than	>
Greater than or equal to	>= or ≥
Equal to	== or =
Not equal to	!= or <>

Integer

Integers are commonly represented internally in a computer system as a group of binary digits, called *bits*. A bit is the smallest unit of data in a computer. A bit has a single binary value, 0 or 1, and bits are stored in multiples of eight, referred to as *bytes*; therefore, there are eight bits to a byte.

The maximum and minimum values of an integer depend on the computer architecture used to run the program and whether the integer is signed or unsigned.

In **32-bit computer systems**, integers that are signed have a minimum value of -2^{31} and a maximum value of $2^{31}-1$; from $-2\,147\,483\,647$ to $2\,147\,483\,647$. Unsigned integers have a minimum value of 0 and a maximum of $2^{32}-1$ ($4\,294\,967\,295$).

In **64-bit computer systems**, integers that are signed have a minimum value of -2^{63} and a maximum value of $2^{63}-1$. Unsigned integers have a minimum value of 0 and a maximum of $2^{64}-1$.

It is important to know the computer architecture on which a program will run before designing and developing a software solution, as going beyond the maximum and minimum values of integers can result in an **integer overflow**. This may result in a program crashing or producing inconsistent or invalid output. Integer overflows compromise a software solution's reliability and security.

Floating point

Floating point numbers, also referred to as 'floats', are the computer representation of real numbers – that is, numbers that allow for decimal places.

Floating point numbers consist of two main parts:

- a *mantissa*, which contains the digits of the number that is represented. These can be either positive or negative
- an *exponent*, which helps determine where the decimal point is placed within the significand.

Two basic formats used for floating point numbers in computer systems are single precision and double precision. Single precision is used in 32-bit systems and double precision in 64-bit systems.

Date/time

The way computers calculate the date is from a baseline starting point. Microsoft Excel defined the start date (and time) as zero at 01-01-1900, or January 1, 1900 (mm-dd-yyyy), with time as a decimal for each day, starting at zero at midnight.

The power of 31 for signed integers represents 32 bits, minus 1 bit that is needed to determine if the signed integer is positive or negative. The 1 that is subtracted from the total is due to computer systems counting from 0, rather than 1.

To discover today's date, enter in any Excel cell, `=TODAY()`. Be sure to set the format to General rather than Date.

This text was written at 4.30 p.m. (1630 hours) on 25 October 2024. Entering `=NOW()` function calculated 45590.690578703700, which is the time and date to the exact second.

An added complication is that the year 1900 was not a leap year, though early programmers assumed it was. Unix time measures time from 1 January 1970.

Example: The time for the Melbourne Cup in 2025 is 3 p.m. on 4 November 2025, or 45965.625. This consists of $\text{=DATE}(2025,11,04)+\text{TIME}(15,0,0)=45965 + 0.625$.

Text

The text data type consists of characters and strings.

Character

The **character** data type is a symbol that has meaning. It can consist of any single meaningful unit, such as a letter, a number, a punctuation mark, a symbol, or even a space. For example, the word ‘example’ has seven characters.

What is determined to be ‘meaningful’ relies on something referred to as **character encoding**. Character encoding is one way a computer program can translate binary data into meaningful characters. There are many character-encoding schemes that handle different character sets. For example, ASCII is a character-encoding scheme that represents English characters, punctuation and numbers. UTF-8 encoding is a character-encoding scheme that can represent characters from other languages, such as Japanese Kanji and Korean Hanja, as well as punctuation symbols such as those representing euro (€) and yen (¥).

String

A set, or sequence, of characters is referred to as a **string**. For example, the string ‘I like pie’ consists of eight characters that are letters, and two characters that are spaces. Strings are often implemented in programming languages as an array of characters.

Characters and strings can undergo the same comparisons as the numeric data type, as listed in Table 3.2. Depending on the programming language used, they can also undergo some of the fundamental data type operations as listed in Table 3.1, such as addition, multiplication and assignment. For example, when working with strings, addition acts to concatenate (add) strings together, which is useful when constructing strings that must incorporate variables. An example of this can be seen in Figure 3.7.

```
1 Input firstName
2 Print "Hello " + firstName + "!"
```

Boolean

Boolean data types have only two possible values: 1 and 0. These are often referenced with the words ‘True’ and ‘False’ in a programming language. This data type is named after George Boole, a 19th-century mathematician who was the first to define an algebraic system of logic. Boolean data types are very useful for systems that require decisions to be made or conditions to be met.

Much like integers, Boolean values can be treated mathematically, allowing for comparison operators such as those listed in Table 3.2. This allows for Boolean logic operations to occur in any programming language.

Boolean values can also be used with the fundamental operators *and*, *or* and *not* in statements where a condition must be met. For example, if a program is required to turn on a light in a room if it is dark and the light is not already on, it could use two Boolean values to test for this condition (Figure 3.8).

Research ‘Y2K bug’. At the turn of last century, when the calendar progressed from 1999 to 2000, many computer systems operated with a two-character year field. This meant that 1900 and 2000 would be confused and date calculations suddenly became negative. This was a major problem. Discover three of the far-reaching unintended consequences of attempting to minimise the year field to two characters. Why is this no longer a problem? What has changed? Are there other impending date/time bugs? (Hint: there are!)

THINK ABOUT APPLIED COMPUTING 3.4

How many character encoding sets are there? Why do you think so many separate encoding sets have been created?

FIGURE 3.7 Pseudocode example of string addition

Boolean values take up only a small amount of space in memory, so it is tempting to use them to store any type of data that seems to have only two values. It is important that you consider future expansions to programs before making a decision that will limit a data type. For example, many old systems that had gender stored as a Boolean value are now having to be rewritten to change gender to a character data type.

FIGURE 3.8 Pseudocode example of Boolean test conditions

```
If isDark = True and lightOn = False Then
    turnOnLight ()
EndIf
```

Some programming languages, such as Python, allow for the use of lists as well as arrays. Lists behave like arrays but can contain more than one data type.

Types of data structures

A **data structure** is a method of organising data to allow particular operations to be performed on them efficiently; in this way, data structures are more complex than data types. The types of data structures used in Applied Computing are: one-dimensional arrays, lists, records, fields and classes.

Array

An **array** is a data structure that contains groupings of data. These elements are all of the same data type, such as character, numeric or Boolean. Arrays can also store groupings of other data structures, such as fields or records, or even other arrays. Arrays are very useful in programming, as they allow for related sets of data to be organised and ordered efficiently.

The simplest array is a list, or a one-dimensional array. The length of the array is the number of items in the list. For example, a teacher might collect data related to the height of each student they teach and store this data as a one-dimensional array of floating point numbers. This is a much more efficient method of storing data than creating a variable to store each student’s height, as it allows for faster sorting and searching.

The contents of an array are referenced using an index value – often an integer starting at 0. The way an array is stored means that each element has a set position within it; this allows for quick access to a particular element of the array, without the need to necessarily check every element.

Arrays can typically use the operations shown in Table 3.3.

TABLE 3.3 Array data structure operations

Add or Append	+
Remove or Delete	-

FIGURE 3.9 Pseudocode example of an array

```
arrayHeights ← [ 1.23, 1.35, 1.21, 1.61 ]
firstStudent ← arrayHeights[0]
secondStudent ← arrayHeights[1]
fourthStudent ← arrayHeights[3]
```

Consider the array in Figure 3.9. In this example, firstStudent would contain the floating point number 1.23, secondStudent would contain 1.35 and fourthStudent 1.61.

In **pseudocode** (covered on pages 124–5), arrays are sometimes indexed starting at 1, but it should always be made clear in the pseudocode comments if this is the case.

List

Lists are an example of a container, as they contain other values, which includes the Null or empty list.

In computer science, a list has a finite number of ordered values, with repetition, where every value is a separate item. They differ from arrays as they can contain elements with different data types.

Record and field

A **record** is a basic data structure for collections of related elements. These elements may or may not be of the same data type. Most frequently, records are used in database systems but they are also commonly used in programming languages where they are referred to as *structs*.

A record consists of a number of **fields** that are typically fixed – that is, the fields do not tend to change once the record is defined and used. Each field has a name and each has its own data type.

For example, a customer record may contain fields such as `firstName`, `familyName` and `dateOfBirth`.

Records are most useful when variables in a collection are related to each other. It provides a logical method of ordering data within a program so that data can be accessed quickly.

In object-oriented programming languages, a record is essentially an object that has no object-oriented features, containing only collections of fields and values. Records and fields can also exist in some types of structured plain text files.

As records contain programmer-defined fields, there are no set operations that can be listed for the record data structure. Rather, there are common operations that can be performed on the record and the fields within it, such as assignment and comparison as well as adding or removing fields.

Representing designs

Once a software requirements specification has been completed, it is important that considerable time is spent designing the software that is going to be written. This helps reduce the time and effort that goes into writing the software, as problems are normally resolved before any code has been written; there is nothing worse than needing to rewrite code due to an issue that could have been resolved in the design stage!

Some common methods of representing designs are to use mock-ups, pseudocode and flowcharts. Each of these methods has a different purpose in the design stage.

Mock-ups

If software will be used directly by people (rather than running hidden deep in the OS), it needs an interface – a place where people can control the program, enter data and receive output. A successful interface must be carefully designed to make it usable and clear.

To design an interface, use a **mock-up**, which is a sketch showing how a screen or printout will look. A mock-up should typically include the following features:

- the positions and sizes of controls such as buttons and scrollbars
- the positions, sizes, colours and styles of text such as headings and labels
- menus, status bars and scrollbars
- borders, frames, lines, shapes, images, decoration and colour schemes
- vertical and horizontal object alignments
- the contents of headers and footers.

A mock-up can be considered successful if you can give it to another person and they can create the interface without needing to ask you questions.

In VCE Applied Computing, it is not mandated that you use software to create your mock-ups. You may use software if you wish, but you may also create them by hand using pen and paper.

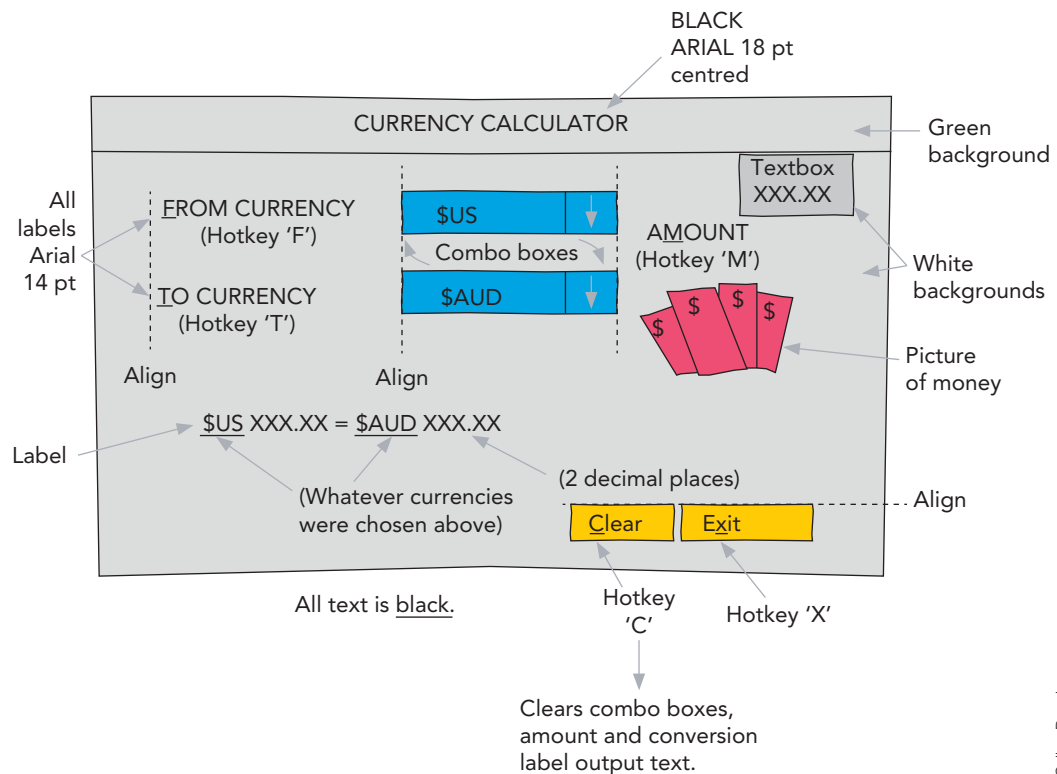


FIGURE 3.10 Mock-up of a screen interface

Selina Dennis

Pseudocode

Writing an algorithm in source code is slow. An algorithm written in source code also limits itself to use in only one compiler. Pseudocode, also known as Structured English, is a quick, flexible and language-independent way of describing a calculation strategy – halfway between English and source code. Once the algorithm is described in pseudocode, it can be converted into source code for any desired programming language.

A good algorithm can be extremely valuable and bring forth great change. A clever strategy can make software run twice as quickly or use half the amount of RAM. An ingenious idea can lead to the development of a program that was once considered impossible. For example, Google's PageRank completely changed the way the world searched the internet, and made billions of dollars for its inventors in the process. The invention of public key encryption finally cracked the age-old problem of how to encode and transmit secrets without having to also send an unlocking key that could be intercepted.

The pseudocode shown in Figure 3.11 determines if a year is a leap year.

```

IF (year is divisible by 4 AND not divisible by 100)
OR (year is divisible by 4 AND 100 AND 400) THEN
    it's a leap year
ELSE
    it's not a leap year
ENDIF
  
```

FIGURE 3.11 Pseudocode to determine a leap year

THINK ABOUT APPLIED COMPUTING

3.5

What are the rules of pseudocode? Easy: there are none. As long as the intention of the calculation is clear, it is good pseudocode. If not, it is bad.

In pseudocode, make sure that you specify assignment (the storage of a value) using the $\leftarrow$ symbol rather than the equals sign (=) that is used in algebra and in most real programming languages – for example:

```
isLeapYear  $\leftarrow$  True
```

The equals sign is reserved for logical comparisons, such as:

```
IF B=0 THEN SoundAlarm()
```

Common features found in pseudocode include:

- iterations/loops, such as **WHILE/ENDWHILE** and **FOR/NEXT**
- condition control structures, especially **IF/ELSE/ENDIF** blocks
- logical operators – **AND**, **OR**, **NOT**, **TRUE** and **FALSE**
- arrays, such as Expenses[3]
- records and fields, such as Customer.firstName, where Customer is the record and firstName is the field
- arithmetic operators (+ – * /) and the familiar order of operations, BODMAS.

Pseudocode punctuation and the names of key words are largely up to you provided it is clear what you mean; for example, it does not really matter if you prefer **WHILE/WEND** or **WHILE/ENDWHILE**.

To ‘get data from keyboard’, you could use **INPUT**, **GET**, **FETCH** or another keyword. To read data from a disk file, you could choose **INPUT**, **GET**, **READ** or something else. To avoid ambiguity, you could explain your pseudocode’s conventions using comments within the pseudocode. Comments can be prefixed with a hash (#) or two forward slashes (//), or included in braces/curly brackets ({}).

```
.....
# GET reads the keyboard.
# READ loads data from a disk file.
# DISPLAY shows output on screen.
# WRITE saves output to a file.
DISPLAY "What is your name?"
GET UserName
OPEN FILE "Users.txt"
READ data for UserName
IF new data exists THEN
    WRITE new data to file
ENDIF
.....
```

FIGURE 3.12
Pseudocode to write
usernames to a file

Flowcharts

Visual representation of a program can assist understanding the logical structure and flow of data. Flowcharts can contain general or specific level of detail about the operation of the logic of the program.

A flowchart is sequential. It has a start and an end, with an order of instruction execution. Loops are a means of representing choices or branching decisions.

Flowcharts and mock-ups can be made using software tools, such as LucidChart and Visual Paradigm.



Weblinks

Lucid Chart

Visual Paradigm

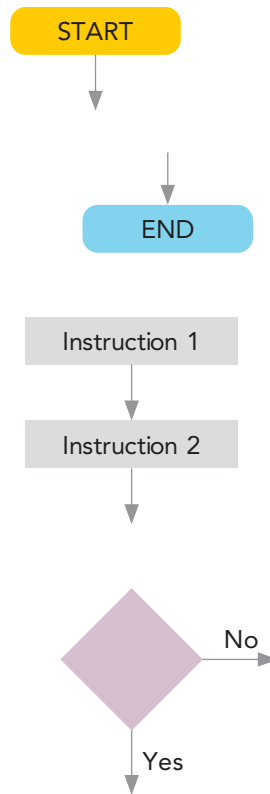


FIGURE 3.13 Flowchart symbols indicate role.

Example: Break down the steps involved in making a cup of tea. The level of detail can vary, and some instructions could have their own flowchart. Boiling water involves kettle, water level, plug in kettle, etc.

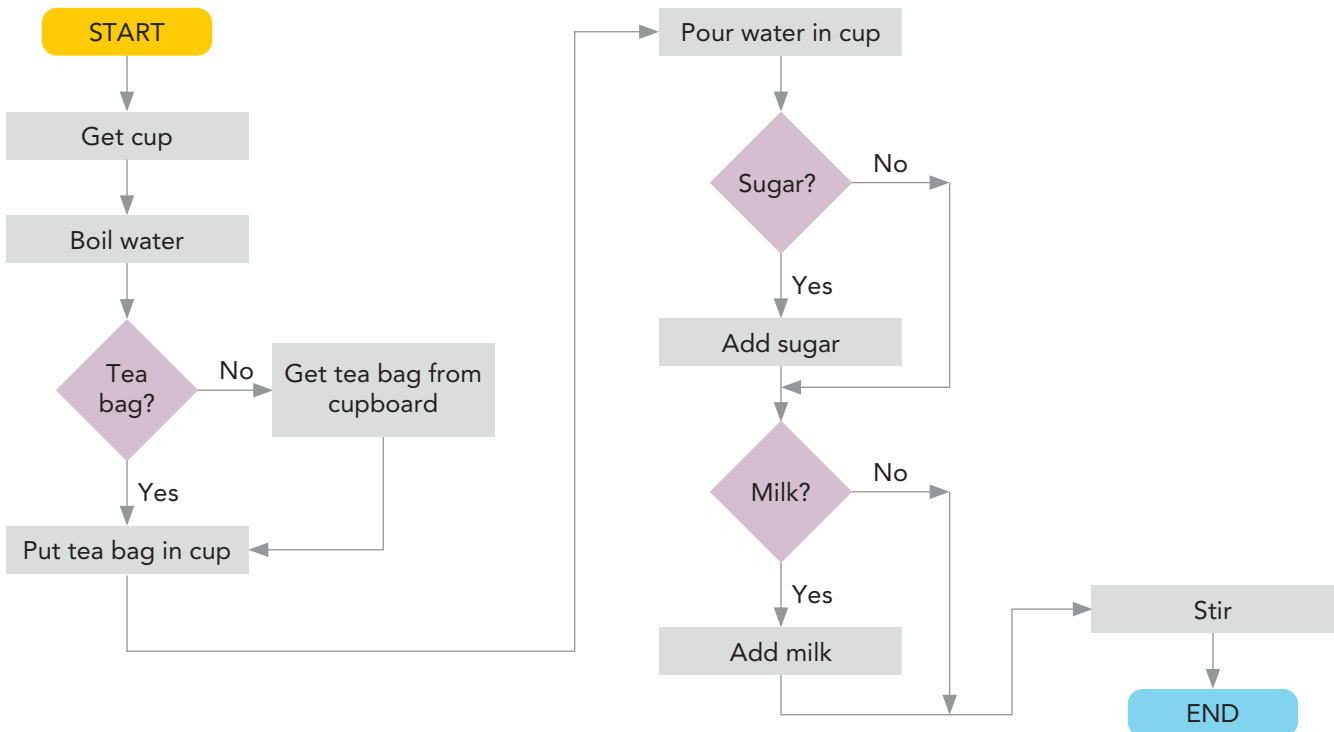


FIGURE 3.14 Simple flowchart for making a cup of tea

3

CHAPTER SUMMARY

Essential terms

32-bit computer system a computer system with 32 bits of memory addresses

64-bit computer system a computer system with 64 bits of memory addresses

array an indexed set of related elements; all elements within an array have a numeric index and are of the same data type

Boolean a logical data type; Boolean data can hold one of only two possible values, true or false

character a data type representing a single letter, number, symbol, punctuation mark or space

character encoding code that allows a computer program to interpret binary digits (0s and 1s) into meaningful units representing characters and numbers. ASCII, UTF and Unicode are types of character encoding.

compiler a program that turns source code into machine language that can be executed by a computer processor

constraint a restriction on what can and cannot occur in the creation of a software solution, external to the solution itself

data structure a method of organising data to allow particular operations to be performed on them efficiently

data type different forms that variables and data may take, which determine the data that a variable may contain, and how the data or variable may be manipulated

debugger a tool used to identify errors in computer software

economic constraint a constraint on the creation of a software solution that relates either to time or budget, or to both

editor a software application that allows programmers to write, edit and manage source code

field a single data item in a record; e.g. FamilyName

fit for purpose well suited for a role or purpose

floating point number computer representation of real numbers, with decimal places

functional requirements requirements of a digital solution that are directly related to what the solution will do

graphical user interface (GUI) a form of user interface that allows users to interact with an application or piece of software

high-level programming language a programming language with strong abstraction from computer hardware, focusing on logic and user-friendly syntax

integer a data type representing whole positive and negative numbers; a number without a fractional or decimal component

integer overflow a condition occurring as a result of a mathematical operation where the output exceeds the maximum or minimum integer value that can be stored on that computer system

integrated development environment (IDE) software that provides tools to programmers to aid in programming, such as source code editing, syntax highlighting, code completion, debugging aids, or tools to help construct a user interface

intellectual property rights legal protections for creations of the mind, including inventions, literary and artistic works, designs and symbols

interface within software, the place where people control the program, enter data and receive output

interpreter a computer program that directly executes source code without needing to have it compiled beforehand

list a data structure that represents a sequence of elements

low-level programming language a programming language that is heavily reliant on the architecture of a particular computer, with commands closely linked to processor instructions

maintainability the extent to which a solution is easy to look after once it has been put in place

mock-up a sketch showing how a screen or printout will look, which is used to aid in the design of an interface

non-functional requirements appearance characteristics that the user or client would like the solution to have; often tied to solution constraints

non-technical constraints factors that may limit or restrict solution development, specifically related to legal, social, usability or economic factors

numeric a data type consisting of whole numbers, referred to as integers, and decimal numbers, referred to as floating point numbers

object any instantiated class that a program can inspect and/or change, in terms of appearance, behaviour or data

platform a combination of operating system (OS) and central processing unit (CPU)

portability the extent to which a solution can easily be used in different operating environments

properties characteristics such as width, colour, visibility

pseudocode code that designs algorithms in a clear, human-readable, language-independent format

record an indexed set of related elements; all elements within a record have a field index and may be of different data types

reliability how much a solution can be depended on to function as designed, and for how long

robustness how well a software solution responds to errors that occur when the software is being used

scope boundaries or parameters of the solution – what it will do and what it will not do

scope creep when a client changes the scope of a project (by increasing it) during the life cycle of the project

source code the human-written and human-readable version of a program

string a data type representing a set or sequence (collection) of characters

syntax the structure of statements in a programming language

technical constraints constraints related to the hardware and software available for a project

uptime time during which a machine, solution or application is operational

usability the extent to which a system is easy to learn and use

Important facts

- 1** Programming languages give instructions to computer processes so they can calculate useful information and carry out tasks.
- 2** To develop software, you need an editor, a compiler or interpreter and a debugger.
- 3** Integrated development environments (IDEs) are commonly used for developing software.
- 4** Emerging trends in programming languages and AI-based code generators have begun to transform how we create software.
- 5** Low-code development involves application creation with minimal hand-coding. Visual interfaces and drag-and-drop tools are used to speed up the development process.
- 6** AI-based code generators enhance the efficiency and quality of code by automatically generating code snippets or entire functions based on given requirements or contexts.
- 7** For a problem to be satisfied by a software solution, the following elements must be provided: scope, functional requirements and non-functional requirements.
- 8** Scope outlines the boundaries or parameters of a solution so all stakeholders are aware of what the solution will contain and what it will not contain.
- 9** Functional requirements relate directly to what the software will do.
- 10** Non-functional requirements relate to the characteristics of the software that are desirable, typically related to the quality of the system produced.
- 11** Non-functional requirements include usability, reliability, portability, robustness and maintainability.
- 12** Solution constraints are factors that may limit or restrict solution requirements. These are determined during the analysis stage of the problem-solving process. Constraints can be categorised as economic, technical and non-technical.
- 13** Data types include numeric (integer, floating point, date/time), text (character, string) and Boolean.
- 14** Data structures include array, list and record.
- 15** Some common methods of representing designs are to use mock-ups, pseudocode and flowcharts.
- 16** Mock-ups are sketches that show how a screen or application will look.
- 17** Pseudocode, also known as Structured English, is a quick, flexible and programming language independent method of describing calculation strategies.
- 18** Flowcharts provide a sequential method of demonstrating the order of instruction execution. They assist in understanding the logical structure and flow of data.



TEST YOUR KNOWLEDGE

Programming and scripting languages

- 1 What is the difference between a high-level programming language and a low-level programming language?

Software development tools

- 2 What are the essential tools used to develop software?
- 3 Provide an example of source code in a programming language of your choice.
- 4 What is the difference between a platform and an operating system?
- 5 Why would a programmer choose to use an IDE?

Emerging trends in programming and AI

- 6 Explain the concept of low-code development platforms. How do they streamline the development process and who can benefit from using them?
- 7 Discuss the importance of readability and maintainability in programming. How do modern programming languages and tools contribute to these aspects?
- 8 Provide an example of a low-code development platform and an AI-based code generator. Explain how each of these tools can be used to enhance the efficiency and quality of software development.

Determining scope

- 9 Define scope.
- 10 What is scope creep?

Functional and non-functional requirements

- 11 What is the difference between a functional requirement and a non-functional requirement?
- 12 Categorise the following as functional or non-functional requirements. If the requirement is non-functional, provide its type.
 - a Customer data should be saved to a file.
 - b Users should only need an hour to be trained fully on the system.
 - c Uptime should be 95 per cent during summer months.
 - d The system should reject invalid dates, such as 30 February.
 - e Jobs should be able to be loaded into a timetable.
 - f The system should stay running, with fewer than five hours per month spent maintaining the system.
 - g All orders should be printed in a report at the end of a work day.



Constraints

- 13 When are solution constraints determined?
- 14 What do solution constraints typically involve?
- 15 Categorise the following solution constraints according to their type.
 - a The system must be completed by the end of the year.
 - b Data entry can occur via touch screen and joystick controller.
 - c The user interface must be usable by colourblind users.
 - d All data covered under copyright laws must be encrypted.
 - e New system equipment must cost no more than \$10 000 in total.

Data types and data structures

- 16 Select the most appropriate data types and structures for the following data.
 - a Inventory: { “socks”: 30, “tables”: 34, “swing sets”: 12, “curtains”: 4 }
 - b 137
 - c False
 - d 5.99
 - e]
 - f potato
 - g 0400999888
 - h 1
 - i Customer: { Dillon, Corp, 25/07/1993, 123 Fourth Street, Fifthsville, VIC, 3000 }
 - j +
 - k Players: { Wajee, Sharon, Penny, Lola, Darius, Shehara, Ling, Rajkumar }
- 17 Reformat this date/time serial number to determine the exact time.
45590.690578703700
- 18 What is the serial number of today?
(Hint: Use Microsoft Excel by entering =NOW() and change the format to General or Number.)
- 19 When would you use a record rather than an array to store a collection of related values?

Representing designs

- 20 When would you use a flowchart over pseudocode?
- 21 What is pseudocode and what is its purpose?
- 22 What is the difference between $\leftarrow$ and = in pseudocode?
- 23 Draw a flowchart for solving a general problem.



APPLY YOUR KNOWLEDGE

Programming practice

- 1
 - a Write a pseudocode program that asks a user for their date of birth. From their input, calculate their current age in years.
 - b Now calculate their exact age in years, months and days.
- 2 ELIZA was an early natural language processing computer program created in the mid-1960s. It simulated conversation by using pattern matching and substitution methodology to respond to user input. Your task is to create a basic version of ELIZA, where the chatbot will interact with the user, providing responses based on their input. This project will help you practise string manipulation, conditional statements and user input handling.
 - a Implement a basic program that prompts the user for input and responds with a fixed message, such as “Tell me more.”
 - b Enhance the program to analyse the user’s input for specific keywords and respond accordingly. For example, if the input contains “sad”, respond with “I’m sorry to hear that. Why are you feeling sad?”
 - c Extend the program to recognise patterns in the user’s input and substitute parts of the input into the response. For example, if the input is “I am feeling [emotion]”, respond with “Why are you feeling [emotion]?”
 - d Allow the user to continue the conversation until they choose to exit. Implement a condition to break the loop, such as the user typing “bye”.
 - e Create a mechanism to store the user’s previous inputs. Use an array or another suitable data structure to keep track of the conversation history. Ensure that this history is updated with each new user input.
 - f Develop logic to use the stored conversation history to provide more meaningful responses. For example, if the user mentions a feeling or an event in a previous input, refer back to it in future responses to show continuity in the conversation.

Programming task: Guessing game

- 3 Design and develop a program that will play a game with a human. The human thinks of an integer between 1 and 100. Your program will use the most efficient strategy possible to guess the number. After each guess, the human must confirm whether the guess was correct, or higher or lower than the secret number.
 - a Write pseudocode to describe the processing strategies involved in developing the program.
 - b Describe the functional requirements of the program.
 - c Design the program’s interface with a mock-up.
 - d Develop the solution, using good naming conventions and internal documentation.
 - e Test the solution and fix all bugs.

Developing software

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

- KK1.2.7: principles of object-oriented programming (OOP), such as: abstraction, encapsulation
- KK1.2.8: features of a programming language
- KK1.2.9: naming conventions for solution elements
- KK1.2.10: purposes of internal documentation
- KK1.2.11: validation techniques for data
- KK1.2.12: debugging and testing techniques for checking software solutions function correctly.

Victorian Curriculum and Assessment Authority,
Victorian Certificate of Education Applied Computing
Study Design 2025–2028, VCAA, 2024, p. 28

FOR THE STUDENT

This chapter relates to VCE Applied Computing Unit 1, Area of Study 2: Programming. It introduces basic programming concepts such as variables, operators and control structures. You will learn about the principles of object-oriented programming, as well as appropriate methods of internal documentation. Later in the chapter, you will begin to understand testing and debugging techniques and the tools programmers use to help debug software, such as creating test data and constructing trace tables. You will continue to learn about the specific programming language that you will use to develop a response to a design brief for Unit 1, Outcome 2. This will involve the creation of a software solution in response to teacher-provided solution requirements. You will need to design and develop a software solution as well as apply suitable validation, testing and debugging.

This chapter continues from Chapter 3, introducing you to the core features of programming that are required for Unit 1, Outcome 2. It does not assume knowledge of any actual language. Instead, pseudocode is used where relevant. Unit 1, Outcome 2, requires that you apply all stages of the problem-solving methodology to design and develop a software solution in response to a set of solution requirements. The solution requirements should include functional and non-functional requirements, constraints and factors that determine scope. The requirements will allow you to draw on knowledge of basic programming concepts, such as variables, control structures, operators, procedures, functions and methods.



Processing features of a programming language

All modern programming languages share the same types of features, such as the ability to store data within variables and execute **instructions** within control structures. While the syntax of individual programming languages may vary, the fundamental logic behind these processing features remains the same.

Variables

Variables are methods of storing data so that they can be retrieved later within a program. Without variables, it would be impossible to reference data once it has been stored in memory. A variable is typically used to store a data type or structure, but it can also be used to store a **pointer** to a function or method. Variables should be named appropriately, following consistent naming conventions.

Initialisation of variables

Initialisation refers to the process of assigning an initial value to a variable. In many programming languages, this is done at the time of variable declaration. The syntax used to initialise a variable can vary depending on the programming language used. Table 4.1 shows how to initialise variables in some programming languages.

TABLE 4.1 Initialising variables in some programming languages

Language	Example
Python	<code>x = 10</code>
Java	<code>int x = 10;</code>
C++	<code>int x {10};</code> <code>int x = 10;</code>
Pascal	<code>x := 10</code>
Lisp	<code>(setq x 10)</code>
Haskell	<code>let x = 10</code>

Accessing variables

Accessing a variable means retrieving the value stored in it. You can access a variable simply by using its name in an expression or a statement – for example, `print(x)` in Python, or `System.out.println(x)` in Java.

Storing data in variables

Storing data in a variable involves assigning a new value to it. This can be done using the assignment operator (the equal sign '=' in many languages). You can see examples of this in Table 4.1.

Arithmetic, logical and conditional operators

Arithmetic, logical and conditional operators are fundamental concepts in programming, each serving a distinct purpose in manipulating data and controlling the flow of execution in code.

Arithmetic operators

Arithmetic operators are used to perform mathematical operations. They are similar to the operations performed in basic mathematics and are used to manipulate numerical values. The most common arithmetic operators are shown in Table 4.2.

TABLE 4.2 Arithmetic operators

Operator	Description	Example	Result if a=10, b=5
+	Addition	a + b	15
-	Subtraction	a - b	5
*	Multiplication	a * b	50
/	Division	a / b	2
%	Modulus (Remainder)	a / b	0

Logical operators

Logical operators are used to perform logical operations, primarily in selection statements, to determine the truthfulness of an expression. They are used to combine or invert **Boolean** values (true or false). The primary logical operators are shown in Table 4.3.

TABLE 4.3 Logical operators

Operator	Description	Example	Result if condition1 = true, condition2 = false
&&	Logical AND	condition1 && condition2	false
	Logical OR	condition1 condition2	true
!	Logical NOT	!condition1	false

While Table 4.3 displays the most common syntax for logical operators in programming languages, these are not the only symbols used. For example, Python allows for the word 'not' to be used to indicate 'not', Visual Basic uses '<>', Prolog uses '\+' and MATLAB uses '~'. It is best to look at the documentation for your language of choice to check the appropriate syntax.

Conditional operators

Conditional operators are used to compare two values or expressions. They are fundamental in control structures such as selections and iterations, as they help in decision-making processes by evaluating conditions. The most common conditional operators are shown in Table 4.4.

TABLE 4.4 Conditional operators

Operator	Description	Example	Result if a=10, b=5
==	Equal to	a == b	false
!=	Not equal to	a != b	true
>	Greater than	a > b	true
<	Less than	a < b	false
>=	Greater than or equal to	a >= b	true
<=	Less than or equal to	a <= b	false

There are also more complex logical operators that you are unlikely to come across in VCE Applied Computing, some of which include XOR, XNOR, NAND and NOR. These would be out of scope of the study design.

Note that there are different symbols used to represent operators in code compared to how you would represent them in pseudocode.

Control structures

There are three fundamental control structures in programming: sequences, conditional statements and iterations.

Sequence

A **sequence** is a set of instructions that executes line by line, a little bit like a recipe; every line of code in the sequence is run in the order that it is written.

In the algorithm shown in Figure 4.1, a sequence of code runs that asks a user for their name, reads the name as input and then greets the user by name. Each line of code in the algorithm is run, in order, only once.

Selection

Selective control structures (selections) allow the application to make decisions during runtime, dependent on given conditions. The code within a **selection statement** can contain instructions, nested conditional statements and iterative structures.

```

ALGORITHM askName()
BEGIN
    PRINT "What is your name?"
    INPUT name
    PRINT "Hello, " + name + ". Nice to meet you."
END
  
```

FIGURE 4.1

Pseudocode example of a sequence of instructions

Selection statements are Boolean, in that they are run based on the result of a condition being evaluated as either True or False. If the condition evaluates as True, the code within that condition is executed. If it evaluates as False, it is not.

The simplest type of selection statement is one that tests against a single condition, such as seen in Figure 4.2.

```

ALGORITHM printPositive()
BEGIN
    INPUT intNumber

    IF intNumber > 0 THEN
        PRINT "The number is positive."
    ENDIF
END
  
```

FIGURE 4.2 Single condition selection control structure

Alternative execution

Another form of selection statement involves an **alternative execution**. This means that if the condition is not met, an alternative block of code will be run. For example, in Figure 4.3, a user is asked if they like pie. If they respond with 'yes', they receive a happy comment. Alternatively, if the user does not input 'yes', a sad comment will be printed instead. Note that the user does not need to input 'no' for the sad comment to be printed – any input aside from 'yes' will execute the alternative code.

```

ALGORITHM likePie()
BEGIN
    PRINT "Do you like pie?"
    INPUT likePie

    IF likePie = "yes" THEN
        PRINT "Hooray!"
    ELSE
        PRINT "That makes me sad. :("
    ENDIF
END

```

As you can see in comparing Figure 4.2 with Figure 4.3, the use of ELSE in a selection statement is optional.

FIGURE 4.3
Selection control structure with alternative execution

Selections with more than one logical expression

A selection statement is not limited to testing only one logical expression; in fact, the number of logical expressions a single selection statement can test is theoretically unlimited. Figure 4.4 contains pseudocode that uses the logical operator AND within a single selection statement to check for two conditions to be simultaneously true.

```

ALGORITHM likePieCake()
BEGIN
    PRINT "Do you like pie?"
    INPUT likePie
    PRINT "Do you like cake?"
    INPUT likeCake

    IF likePie = "yes" AND likeCake = "yes" THEN
        PRINT "Hooray!"
    ELSE
        PRINT "That makes me sad. :("
    ENDIF
END

```

FIGURE 4.4 Simple selection statement with more than one logical expression

When writing selections with more than one logical expression, it can be useful to construct a **truth table** to make sure that no logic errors have been included. Truth tables use Boolean algebra to test each combination of values in a condition. For example, Table 4.5 contains a truth table to check the logical expressions in the pseudocode from Figure 4.4. Because the logical operator connecting the two conditions is 'AND', both conditions need to be true for the conditional statement as a whole to evaluate as true, which is shown when only a single case in the truth table evaluates the whole condition as True.

TABLE 4.5 Truth table to evaluate AND

likePie	likeCake	likePie AND likeCake
True	True	True
True	False	False
False	True	False
False	False	False

If the logical operator connecting the two conditions was 'OR', the resulting truth table is what can be seen in Table 4.6. In this instance, the use of 'OR' expands the number of cases where the condition would evaluate to True to three: the only time it evaluates to False is when both conditions are False.

TABLE 4.6 Truth table to evaluate OR

likePie	likeCake	likePie OR likeCake
True	True	True
True	False	True
False	True	True
False	False	False

Truth tables are a systematic method of testing the logic of a selection statement. They are particularly useful when there are more than two conditions within a single statement. For example, Table 4.7 contains a truth table for the selection statement in Figure 4.5. The truth table shows there are three cases where the entire selection statement would evaluate to True, and five cases where it would evaluate to False. This would not be easily apparent without constructing the truth table.

FIGURE 4.5 Complex selection statement with more than one logical expression

```

ALGORITHM multiConditions()
BEGIN
    INPUT a, b, c, d
    IF a < b AND ( b < c OR c < d ) THEN
        PRINT "Condition met."
    ELSE
        PRINT "Condition not met."
    ENDIF
END

```

TABLE 4.7 Truth table for Figure 4.5

a < b	b < c	c < d	b < c OR c < d	a < b AND (b < c OR c < d)
True	True	True	True	True
True	True	False	True	True
True	False	True	True	True
True	False	False	False	False
False	True	True	True	False
False	True	False	True	False
False	False	True	True	False
False	False	False	False	False

A truth table can be used in conjunction with testing tables to determine if an algorithm is without logical errors. They can also be used to help select **test data** for testing an algorithm, as discussed later in this chapter.

Chained selection

The algorithm shown in Figure 4.6 uses a more complex set of selection control structures (IF/ELSEIF) in order to react to user input when a condition needs to be tested more than once. This is referred to as a **chained selection**. In this example, the user can select four operations: addition, subtraction, multiplication and division. The program must therefore test the user input four times to see if it matches against the four conditions given.

As the algorithm uses ELSEIF, it will check each selection statement only if the condition before it evaluates as False. Without the use of ELSEIF, each selection statement would be run in sequence regardless of whether the selection before it evaluated as True or False. This is an important characteristic of the selection control structure that programmers often forget, resulting in logical errors in code.

```

ALGORITHM computeOperation()
BEGIN
    PRINT "What is the first number?"
    INPUT firstNumber
    PRINT "What is the second number?"
    INPUT secondNumber
    PRINT "What operation would you like to perform?"
    INPUT operationChosen

    Total ← 0
    IF operationChosen = "add" THEN
        Total ← firstNumber + secondNumber
    ELSEIF operationChosen = "subtract" THEN
        Total ← firstNumber - secondNumber
    ELSEIF operationChosen = "multiply" THEN
        Total ← firstNumber * secondNumber
    ELSEIF operationChosen = "divide" THEN
        Total ← firstNumber / secondNumber
    ELSE
        PRINT "Invalid operation chosen."
    ENDIF

    PRINT "The result is: " + Total
END

```

FIGURE 4.6

Pseudocode example of a chained selection control structure

```

ALGORITHM ifElseExample()
BEGIN
    PRINT "What is the current temperature?"
    INPUT currentTemp

    IF currentTemp < 10 THEN
        PRINT "It is very cold."
    ENDIF
    IF currentTemp < 20 THEN
        PRINT "It is a little cool."
    ELSE
        PRINT "It is very warm."
    ENDIF
END

```

FIGURE 4.7

Pseudocode example of selections with logical errors

Consider the pseudocode in Figure 4.7. Assume the current temperature is input as 9 degrees. The first IF condition tests to see if the temperature is less than 10 degrees – as 9 degrees is less than 10 degrees, the algorithm will print ‘It is very cold’ and then continue in sequence to the next line of code in the program, which is the second IF condition. The second IF condition will check if the temperature is less than 20 degrees: as 9 degrees is less than 20 degrees, it will print ‘It is a little cool’ and then continue in sequence to the next line of code in the program, which is the ELSE condition. This portion of the code is not run: as the IF statement it is attached to evaluated to True, the ELSE condition has not been met

It is important to use consistent formatting in your source code. This not only makes the code easier to read, it also reduces the chance of logical errors occurring due to nested selections. The pseudocode in Figure 4.8 shows the hierarchy of the conditional statements because of the use of tabs and spacing; it is much easier to read than if no tabs or spacing were used.

FIGURE 4.8

Pseudocode example of a nested selection statement

Most programming languages include a method for exiting an iteration early. These are often referred to as 'breaks'.

FIGURE 4.9

Pseudocode example of a WHILE loop to read from a file

(9 degrees is not greater than 20 degrees). After the selection statements are run, it is clear that a logical error has occurred, as two print statements have been produced instead of one.

There is no limit to the number of conditions that can be contained in a chained selection statement. Note also that chained conditions do not need to contain an ELSE statement.

Nested selection

Selection statements can also be placed inside each other. This type of condition is referred to as a **nested selection statement**. Nested selection statements are useful when multiple conditions must be handled within the code. For example, Figure 4.8 contains pseudocode in which a selection statement is used to check if it is raining. If it is, another check is made to see if the person has an umbrella. If they do, they use the umbrella; otherwise they get wet. The check for 'hasUmbrella' is nested within the check for 'isRaining'.

```
ALGORITHM checkUmbrellaUsage()
BEGIN
    INPUT isRaining
    INPUT hasUmbrella

    IF isRaining = True THEN
        IF hasUmbrella = True THEN
            useUmbrella()
        ELSE
            getWet()
        ENDIF
    ENDIF
END
```

Iteration

An **iteration**, also known as a loop, is used to repeat sections of code multiple times until a condition is met. There are four main types of iterations: WHILE loops, DO/WHILE loops, FOR loops and REPEAT/UNTIL loops.

WHILE loops

A **WHILE loop** is a section of code that is run when, and for as long as, a condition is met. These types of loops are useful when the programmer does not know when the condition might be met – for example, when running sections of code based on user input that will only cease when a user inputs a particular key sequence.

An example of a WHILE loop can be seen in Figure 4.9. This type of WHILE loop is very common as, when opening a file for reading, it is not possible to tell how many lines there are in the file until all those lines are actually read.

```
ALGORITHM readFromFile()
BEGIN
    INPUT fileName

    fileObject ← open filename for reading
    WHILE end of file is not reached DO
        nextLine ← read one line from fileObject
        PRINT nextLine
    ENDWHILE
END
```

Key elements of WHILE loops are:

- They are used when it is unknown how many times the loop will execute, or if it will execute at all.
- The condition being tested within the WHILE loop must be met at least once for the code within it to be executed.
- If the condition being tested within the WHILE loop is always True, the loop will never terminate; this creates an **infinite loop**.

DO/WHILE loops

A **DO/WHILE loop** is similar to a WHILE loop in that it executes code within the loop for as long as a condition is met.

Figure 4.10 shows an example of pseudocode to read the contents of a file using a DO/WHILE loop.

The difference between a WHILE loop and a DO/WHILE loop is that a WHILE loop may not necessarily run if the condition being tested is never True, whereas a DO/WHILE loop always runs at least once. For example, in the pseudocode shown in Figure 4.9, it is assumed that there is at least one line to read in the file that is opened. If this is not the case and this code were to be implemented in a programming language, it would produce a runtime error.

```

ALGORITHM readFromFile()
BEGIN
    INPUT fileName

    fileObject ← open filename for reading
    DO
        nextLine ← read one line from fileObject
        PRINT nextLine
    WHILE end of file is not reached
END

```

Key elements of DO/WHILE loops are:

- They are used when it is unknown how many times the loop will execute.
- The code within the DO/WHILE loop will always execute at least once.
- If the condition being tested within the DO/WHILE loop is always True, the loop will never terminate; this creates an infinite loop.

FOR loops

A **FOR loop** is a section of code that is run a predefined number of times. Such loops are particularly useful to perform an action on every element of an array, or to perform a basic search through a set of elements in a data structure.

FOR loops need three pieces of information to execute. The first is a starting point or counter, the second is the end condition, and the third is a statement called an increment that increases (or decreases) the starting point (counter) so that it approaches the end condition.

Most programming languages have a special format for writing FOR loops. For example, in C and C++ a FOR loop is written as shown in Figure 4.11, and in Visual Basic (VB) it is written as shown in Figure 4.12.

4.1 THINK ABOUT APPLIED COMPUTING

Research infinite loops and find out the different reasons they can occur. Learn the key commands your IDE uses to break out of infinite loops, then find some examples in the programming language of your choice and run them.

FIGURE 4.10
Pseudocode example of a DO/WHILE loop to read from a file

FIGURE 4.11 FOR loop in the style of the C, C++ and C# programming languages

```
for ( starting_point; end_condition; increment )
{
    statement(s);
}
```

FIGURE 4.12 FOR loop in the style of the Visual Basic programming language

```
For counter = start_condition to end_condition
    Statement(s)
Next
```

An example of a FOR loop in pseudocode is shown in Figure 4.13. This FOR loop checks every element of an array to see if a word being searched for is found in the array. Each time the loop is executed, the end condition ($i < iNumNames$) is tested; the loop will continue to run for as long as this returns True.

Key elements of FOR loops are:

- They run for a set number of times and this number must be known beforehand.
- They will only execute the code inside them if the end condition is still being met, and may not execute the code at all.
- As with WHILE and DO/WHILE loops, it is possible for a FOR loop not to terminate; this occurs only if the increment and end condition are unrelated. This type of occurrence would be considered a logic error.

FIGURE 4.13 Example of a FOR loop checking every element of an array

```
ALGORITHM checkArray()
BEGIN
    INPUT arrayNames
    INPUT searchTerm

    iNumNames ← length of arrayNames
    FOR i ← 0, i < iNumNames, i ← i + 1 DO
        IF arrayNames[i] = searchTerm THEN
            PRINT "Found " + searchTerm
        ENDIF
    ENDFOR
END
```

In some programming languages, FOR loops can also be written using an iterator loop, rather than a counter with an incrementor. An example of this type of FOR loop can be seen in Figure 4.14. Notice that a temporary variable, 'eachName', has been created. This is used throughout the FOR loop to refer to the current element in the array. Once the FOR loop has finished executing, the temporary variable is 'deleted' (freed from memory).

FIGURE 4.14 Pseudocode example of a FOR loop using an iterator loop and temporary variable

```
ALGORITHM checkArray()
BEGIN
    INPUT arrayNames
    INPUT searchTerm

    FOR eachName in arrayNames DO
        IF eachName = searchTerm THEN
            PRINT "Found " + searchTerm
        ENDIF
    ENDFOR
END
```

REPEAT/UNTIL loops

Much like WHILE loops, **REPEAT/UNTIL loops** repeatedly run a source code within the loop; they differ, however, in the treatment of the condition that terminates the loop. A WHILE loop will run for as long as a condition returns True, whereas a REPEAT/UNTIL loop will run for as long as a condition returns False. Figure 4.15 demonstrates pseudocode that uses a REPEAT/UNTIL loop to read lines from a file.

```

ALGORITHM readFromFile()
BEGIN
    INPUT fileName

    fileObject ← open filename for reading
    REPEAT
        nextLine ← read one line from fileObject
    UNTIL end of file is reached
END

```

FIGURE 4.15 Example of a REPEAT/UNTIL loop to read from a file

Key elements of REPEAT/UNTIL loops are:

- They are used when it is unknown how many times the loop will execute.
- The code within the REPEAT/UNTIL loop will always execute at least once.
- If the condition being tested within the REPEAT/UNTIL loop is always False, the loop will never terminate; this creates an infinite loop.

Functions

A function is a block of code that is intended to be used repeatedly within an application. The code within a function executes an algorithm, and typically provides a **return value** as a result. To use a function within source code is to ‘call’ it. An example of a **function call** can be seen in Figure 4.16; both useUmbrella() and getWet() are function calls.

```

ALGORITHM checkUmbrellaUsage()
BEGIN
    INPUT isRaining
    INPUT hasUmbrella

    IF isRaining = True THEN
        IF hasUmbrella = True THEN
            RETURN useUmbrella()
        ELSE
            RETURN getWet()
        ENDIF
    ENDIF
    RETURN False
END

```

Notice that functions in pseudocode use parentheses after the name of the function. This helps distinguish them from variables.

FIGURE 4.16 Example of the use of function calls

A **procedure** refers to a set of instructions grouped together to perform a specific task. Procedures are a type of **subroutine** or **function**, often used in **procedural programming languages**. However, unlike functions, procedures typically do not return a value. Instead, they are used to execute actions or a series of operations.

TABLE 4.8 Function declarations in some popular programming languages

C, C++, C#	<code>int max(iNumOne, iNumTwo);</code>
VB, VB.Net	<code>Function max(byVal iNumOne as Single, byVal iNumTwo as Single)</code>
Python	<code>def max(iNumOne, iNumTwo):</code>
PHP	<code>function max(\$iNumOne, \$iNumTwo);</code>
Swift	<code>func max(iNumOne: Int, iNumTwo: Int) -> Int</code>

Many programming languages require functions to be declared before they can be defined. Similarly, functions must be defined before they can be used within the source code. For this reason it is a good idea to place all function declarations at the top of your source code, and to place function definitions near the top as well.

It is always advisable to use a built-in function whenever possible as they have been tested, and are less likely to contain bugs compared to a user-defined function.

Functions require a **function declaration** that names the function and its **arguments**. In many languages, function declarations must include the return value data type. They can also include an optional reference to the **function visibility**.

Functions must then have a **function definition** written, which simply means that the function must be written. Some languages, such as C and C++, require that the function declaration is written separately, before the function definition, whereas other languages, such as Python and VB.Net, include the function declaration as part of the function definition. Once a function is defined and written, it can be used throughout the source code.

Almost all programming languages have **built-in functions** that can be used without needing to provide a function declaration or function definition. These functions have been written by the creators of the programming language to execute common sequences of code, such as drawing a widget on a user interface, printing text to a screen, computing mathematical equations such as square roots and powers, or accepting input from a keyboard.

As well as having built-in functions, almost all programming languages allow you to include additional functions, such as those contained within third-party packages, in your source code. These can be very useful to include so that you are not re-creating code that has already been written and tested.

Arguments/parameters

Functions can optionally require arguments, also known as **parameters**, which act as specific inputs that are ‘passed’ to the function. The data within the arguments passed to a function is assigned a temporary variable name as part of the function declaration. This allows the use of **local variables** within a function and thus avoids the need to use **global variables** to access data that exists outside the function.

Many programming languages have two categories of arguments that can be passed to a function: those that are **pass by reference** and those that are **pass by value**. ‘Pass by reference’ means that the original data being passed into the function can be modified without needing to be ‘returned’. ‘Pass by value’ means that the original data is left unchanged, even if the data in the temporary variable is modified within the function. In those languages that only use pass by value in functions, modifying the data stored in the original variables that have been passed to the function requires that the modified data must be returned back to the source code that called the function.

Figure 4.17 demonstrates how to declare a function using pseudocode. Note that the data types of the arguments, and the function’s return value, are defined within a pseudocode comment. The arguments are included as part of the function definition to distinguish them from other types of input.

THINK ABOUT APPLIED COMPUTING

4.2

Research some of the third-party packages that are available to you in your chosen programming language. Find out how to include these packages in your own code. Most languages require that you use a special instruction such as ‘include’, ‘using’ or ‘import’.


```

FUNCTION max( iNumOne, iNumTwo )
{ Purpose: return max value as an integer }
{ Arguments: iNumOne and iNumTwo are integers }
{ Output: integer, the maximum integer }
BEGIN
  IF iNumOne > iNumTwo THEN
    RETURN iNumOne
  ELSE
    RETURN iNumTwo
  ENDIF
END

```

FIGURE 4.17
Representation of a
function in pseudocode

Object-oriented programming

Object-oriented programming (OOP) is a style of programming that revolves around the use of ‘objects’. These objects are structures that hold data and code: the data is stored as fields (also referred to as attributes or properties), and the code is organised in procedures, commonly called methods. Two key principles of OOP are: **abstraction** and **encapsulation**.

Abstraction

Abstraction in OOP is a way of managing complexity by reducing and simplifying interactions with objects. It involves focusing on the essential qualities of an object in relation to the perspective of the viewer, rather than concentrating on its implementation details. This is achieved by creating classes that represent abstract concepts. These classes define the essential attributes and behaviours relevant to that concept while hiding the unnecessary details from the user.

For example, in a class representing a smartphone, abstraction allows a user to know that they can call, message or take photos, without needing to understand the underlying technology, such as signal processing or sensor operation. Abstraction is about creating a simple model that represents complex entities in a manageable way.

Abstraction with a MusicPlayer class

Consider the example of a *MusicPlayer* class to illustrate abstraction. Abstraction for the *MusicPlayer* class means exposing functionalities that are relevant to the user, such as playing a song, pausing or skipping tracks, while concealing the details of how those functionalities are implemented.

Example: *MusicPlayer* class

Essential features (exposed to the user):

- **play(song):** Plays the selected song. The user does not need to know how the music player accesses the song file, decodes the audio format or handles the audio output to the speakers.
- **pause():** Pauses the currently playing song. The internal state management, such as how the current position in the track is stored, is hidden.
- **skip():** Skips to the next song. The complexity of selecting the next track in the playlist or queue is abstracted away.
- **volumeUp()** and **volumeDown():** Adjusts the volume. The underlying audio processing and hardware interaction are not exposed.

**CASE
STUDY**



Complex details (hidden from the user):

- Audio decoding algorithms
- File I/O operations for reading audio files
- Error handling for unsupported file formats or corrupted data
- Managing audio hardware interfaces and drivers.

The `MusicPlayer` class abstracts the complexity of playing music. Users of the class interact with a clean and simple interface (methods like `play`, `pause`, etc.) and are not burdened with the underlying complexities of audio processing, file management and hardware interaction. This abstraction makes the `MusicPlayer` class easy to use and integrate into other systems, such as a mobile app or a web application, without requiring deep knowledge of audio technology.

Encapsulation

Encapsulation is about bundling the data and the methods that manipulate that data into a single unit, known as a class, and controlling their access in a safe way. It is a protective barrier that keeps the data safe from outside interference and misuse.

Data encapsulation led to the important OOP concept of data hiding. For instance, a class `BankAccount` may have a private attribute `accountBalance` and public methods `deposit()` and `withdraw()` to manipulate this balance. The internal state of the `BankAccount` is hidden from the outside, only allowing interaction through its public methods, ensuring that the balance cannot be changed arbitrarily and maintaining integrity of the data.

One of the key benefits of abstraction is the ability to have some control over if and when data is changed; encapsulation is therefore a useful method of strengthening the security of software. Access modifiers, studied in Unit 4, Outcome 1, will allow you to explore how to add security measures into classes.



CASE STUDY

Encapsulation with a Thermostat class

Consider the example of a *Thermostat* class to illustrate encapsulation.

Example: Thermostat class

Attributes (data):

- `currentTemperature`: A private variable that stores the current room temperature.
- `desiredTemperature`: A private variable that stores the temperature set by the user.

Methods (functions):

- `increaseTemperature()`: A public method to increase the desired temperature. It internally adjusts the `desiredTemperature` attribute.
- `decreaseTemperature()`: A public method to decrease the desired temperature. Similar to `increaseTemperature`, it modifies the `desiredTemperature` attribute.
- `getCurrentTemperature()`: A public method that allows the user to view the current room temperature. It returns the value of `currentTemperature` without allowing direct modification.
- `updateInternalState()`: A private method that checks and adjusts the actual heating or cooling device to reach the desired temperature. This method is not accessible from outside the class.

In the Thermostat class, the actual temperature values (`currentTemperature` and `desiredTemperature`) are encapsulated within the class. They are marked as `private`, meaning they cannot be accessed directly from outside the class. Instead, they are modified through public methods like `increaseTemperature()` and `decreaseTemperature()`. This set-up ensures that the internal state of the thermostat can only be modified in controlled ways, preventing misuse or inconsistent states (like setting an impossibly high temperature).

The `updateInternalState()` method is a crucial part of the thermostat's functionality, but it is hidden from the user. This method might contain complex algorithms to control the heating or cooling system efficiently, but this complexity is encapsulated within the class.

Classes

A class is a feature of object-oriented programming that allows a programmer to group together related functions and variables in one place. This acts as a template for creating user-defined objects.

In a business application, a programmer could write a 'user' class that contains typical user variables and methods, such as `username`, `password`, the ability to log in and the ability to change user details. They could then use this base 'user' class to create an 'administrator' class that adds other methods, such as the ability to change other users' passwords and user details. This is demonstrated in Figure 4.18, where the `Administrator` class inherits all of the variables and methods from the `User` class and then adds three more. These two classes, `Administrator` and `User`, could then be instantiated into objects within the program to allow for two different types of users to exist. This saves programming time, as the basic user methods do not need to be rewritten when creating a different type of user.

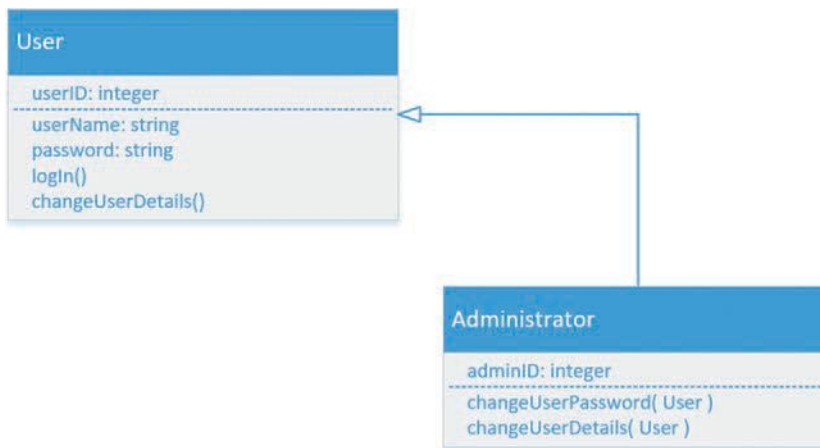


FIGURE 4.18 Example of inheritance

Methods

A **method** is a function or procedure that exists within a class. A special type of method is an **event**, which is a method that is called when an object's state changes; this means that something has occurred to trigger the event. For example, pressing a button in a user interface object can trigger an event to submit that button.

4.3

THINK ABOUT APPLIED COMPUTING

What type of visibility should the functions inside the user and administrator classes have? Why?

4.4

THINK ABOUT APPLIED COMPUTING

Take a look at some of the classes that exist in the programming language you are using. What type of methods are contained within them? As a start, you may want to look at user interface classes (widgets) to see what events they support.

Events can be user-defined or they can be built-in events.

```
ALGORITHM useMethod()
BEGIN
    mathObj = create new object from math class
    mathObj.addTwoNumbers(4, 1)
END
```

FIGURE 4.19 Pseudocode of an object and a method

Programming naming conventions

A **naming convention** is a set of rules that is used when creating variables, subroutines, functions, methods, objects, classes and so on in programming source code as well as in internal documentation. It is a consistent and meaningful way of labelling each of these elements so that they are easily read and understood. The most useful naming conventions allow a programmer to know the purpose of an element and, if relevant, its data type or structure.

Each programming language tends to have a set of language-specific conventions to follow when naming elements. For example, Microsoft.NET, Python and Swift all use a convention called ‘camel case’ within their code. Two other common naming conventions are ‘snake case’ and ‘Hungarian notation’.

Camel case

Camel case, also known as camel caps, uses compound words and phrases as a naming convention, where each word after the first begins with a capital letter – for example, camelCase. No spaces or punctuation are included when naming variables and other elements using camel case. While multi-word variable and function names are useful, it is important that these are as short as possible while remaining meaningful. Often, this is achieved through abbreviating some of the words in the compound phrase. A variable named firstNameOfEmployeeWhoIsPartTime is not as effective as one named firstNameEmplPT as it is too long; writing code using the longer version of this variable would be very tedious.

Snake case

Snake case is very similar to camel case but, instead of compounding phrases into a single word without spaces, it joins each word in the phrase using an underscore – for example, snake_case. All words in snake case are in lower case. Many programmers prefer snake case over camel case as it is easier to read the variable and function names with the underscores separating each word.

Hungarian notation

Hungarian notation, in particular a variety referred to as Systems Hungarian, is similar in style to camel case, in that it compounds words and phrases, without spaces, and each word after the first begins with a capital letter. Hungarian notation additionally adds a *prefix* (that is, an initial letter sequence) before the name of the variable. This letter sequence represents the data type or structure of the variable – for example, iNumEmployees to represent the number of employees as a whole integer. This can be useful when programming using programming languages that are not dynamically typed, as it immediately tells a programmer what data type or structure they are handling when they read the variable name.

TABLE 4.9 Some Hungarian notation prefixes

iNumEmployees	Integer
arrEmployees	Array
strEmployee	String
chEmployeeGender	Character
fpEmployeeHeight	Floating point
fnEmployeeFunction	Function
clCustomer	Class
objCustomer	Object
mGetAllCustomers	Method

Hungarian notation is a flexible naming convention, and many programmers create user-defined prefixes that are meaningful for the programming language they are using, such as some of those used in Table 4.9. For example, while classes and objects were not considered in the initial Hungarian notation prefixes, many programmers use ‘cl’ as the prefix to represent a class and ‘obj’ for an object. With the increasing prevalence of dynamically typed programming languages, however, most programmers prefer to use either the camel case or the snake case naming convention over a modified Hungarian notation.

Internal documentation

One of the most important, yet most easily forgotten, aspects of writing good code is including meaningful **internal documentation**. Internal documentation consists of notes and comments written by a programmer within the code itself. It includes information about the program as a whole, as well as about each of the classes, functions, methods, objects and algorithms within it. It is often combined with meaningful, well-named variables to create manageable and effective code.

Internal documentation has no impact on the compilation or running of the code itself. It exists only to provide context and important information about the code itself. Often it is most useful when a programmer is reading through code that they did not write themselves, or that was written a long time ago. In this situation, internal documentation means that the programmer does not need to rely on remembering what they wrote or on interpreting complex algorithms to understand how the program works. Well-written internal documentation saves time as it reduces effort on the part of the developer, making it a core feature of efficiency in relation to the creation and maintenance of a software solution.

Internal documentation conventions

While there is no single set of conventions for internal documentation, there are many common elements that should be included in internal documentation:

- a **header comment**, containing the name of the file, a brief description of the program, the author’s name, and the date when the program was first created
- documentation of classes and methods, describing their behaviour and how they are used, including any expected inputs and outputs and their respective data types



Weblink
Git/GitHub

- function and subroutine comments, describing their purpose, as well as describing all inputs and outputs and their respective data types
- single-line comments, providing brief summaries of portions of code
- multi-line comments, explaining a complex algorithm within the code itself
- descriptions of how to test aspects of the software
- extra information on upgrades, changes or enhancements made to the program.

While internal documentation can include comments related to revisions and new versions, this does not replace the need to use an effective **version control system**.

Version control is the management of changes to source code files and other project-related documents. This is important throughout the duration of a software project. Version control systems typically run as stand-alone programs or web-based systems that not only help track changes to documents, but also allow more than one developer to work on source code at the same time. They also give a developer the ability to revert back to previous versions of source code. One popular version control system is Git/GitHub (see weblink).

When writing internal documentation, it is important that the comments made within the source code are well formatted to allow them to be easily read. Comments should be separated from code by a blank line before the comment. For multi-line comments, a blank line should be included both before and after the comment. A comment should be vertically aligned with the indent level of the current code to make it clear which code it is associated with. Depending on the programming language used, comments must be enclosed using special characters. Table 4.10 includes a list of the characters used in some popular programming languages.

TABLE 4.10 Common programming language commenting conventions

C, C++, Java, JavaScript, Swift	// this is a single-line comment /* this is a multiple-line comment */
Python	# this is a single-line comment
PHP	# this is a single-line comment // this is also a single-line comment /* this is a multiple-line comment */
XML, HTML	<!-- this is a single-line comment --> <!-- this is a multiple-line comment -->
VB, VB.Net	` this is a single-line comment



Weblink
Doxygen

Many programming languages have tools that make creating internal documentation easier. There are also tools that extract internal documentation to create reference manuals and online documentation for users. One tool that extracts internal documentation is Doxygen, which supports multiple programming languages (see weblink).

While it is important to include internal documentation in all software modules, the comments that are included should be meaningful and non-trivial. Unless it is being used for teaching purposes, internal documentation should not simply state or restate what is occurring in the code, in particular if this is clearly apparent in the code itself.

For example, the commented Python code in Figure 4.20 would be considered trivial as the code comments do little more than restate what the next line of code does.

```
def readFromCSV( csvFileName ):

    # open the CSV file for reading
    csvFile = open(csvFileName, "r")

    # read all of the lines in the csv file
    csvContents = csvFile.readlines()
    return csvContents
```

FIGURE 4.20 Poorly commented code

Better comments would be those seen in Figure 4.21. Notice that the code comments provide information about the function, its inputs and its outputs, including reference to data types.

```
# Function: Reads a CSV file and returns its contents
# Input: String, filename
# Output: Array of Strings, contents of the file

def readFromCSV( csvFileName ):

    csvFile = open(csvFileName, "r")
    csvContents = csvFile.readlines()
    return csvContents
```

FIGURE 4.21 Appropriately commented code

Validation techniques

Validation is the process of checking that input data is reasonable. Validation does not and cannot check that inputs are accurate.

Existence checking

An **existence check** checks whether a value has been entered at all. This is particularly useful to ensure that all required fields in a form have been completed before saving the contents of those fields to a file.

Type checking

A **type check** is a useful method of confirming that the values entered into fields are of the expected data type. It will confirm if the wrong type of data has been entered in fields, such as if strings are entered into fields that expect only integer values.

When implementing type checking, it is important to consider how inputs from a user are processed by the selected programming language. For example, in Python, unless explicitly handled otherwise, all inputs are treated as strings, even if they are numeric. In languages such as these, there typically exists methods for checking type, such as the process of casting the data as a particular data type to see if it remains valid.

Range checking

A **range check** checks that data is within acceptable limits or comes from a range of acceptable values. For example, students enrolling in kindergarten must be between the

ages of 3 and 6 years (acceptable limits). As another example, the product size must be small, medium or large (acceptable values).

```
def validate_input(data):
    # Existence Checking
    if data is None:
        return "Data does not exist."

    # Type Checking
    if not isinstance(data, int):
        return "Data is not an integer."

    # Range Checking
    if data < 0 or data > 100:
        return "Data is out of range. It should be between 0 and 100."

    return "Data is valid."

# Test the function with different inputs
test_data = [None, "string", -1, 0, 1, 99, 100, 101]

for item in test_data:
    print(f"Testing {item}: {validate_input(item)}")
```

FIGURE 4.22 Example validation checks in Python

Testing and debugging techniques

One of the most important steps when building an application is checking to ensure that each module that has been built meets the design specifications that were set out for it.

When checking to see if a module meets design specifications, it is important to make sure it performs as expected with appropriate inputs, that it is usable and efficient, and that it achieves what the design specification outlines. This often involves internal testing, where the programmer tests the program themselves; external testing, where a quality assurance tester tests the program using **test cases** based on the design specifications; and client testing, where the client participates in walk-throughs and reviews of the software to confirm that it is as they specified in the design stage.

In VCE Applied Computing, checking of modules to ensure that they meet design specifications is mostly done through internal testing methods. An important aspect of this is to make sure that the modules that have been built are as bug-free as possible. This is achieved by completing appropriate testing activities, such as establishing test cases that determine test data and **expected results**, conducting tests, recording results, and then correcting any errors.

One method used in the testing process is **debugging**, which is a testing method that includes finding errors through the construction of **trace tables** and testing the system using test data.

Types of errors

Different types of errors can occur throughout the development of a software solution. Each error can be categorised as a **syntax error**, a **runtime error** or a **logic error**.

Syntax errors

Each programming language has a defined syntax, which is a set of rules that define the arrangement and structure of source code within the programming language. These rules also define which symbols and characters can and cannot be used within variable names.

Syntax errors occur while writing code and are typically fixed immediately, as they prevent code from compiling. These are errors such as missing brackets or semicolons, not passing the correct number of arguments into a function, or not putting quotes around characters and strings. Most IDEs can identify syntax errors during development, while the programmer is writing the code, and most **compilers** will indicate on which line a syntax error occurs, which allows a programmer to find the error easily.

Runtime errors

Unlike a syntax error, a software module with a runtime error will compile without any error notifications. It is not until the program is run and used that a runtime error can appear. These types of errors often result in the program crashing or printing error messages. For example, one type of runtime error is a **memory leak**, where a program continually uses more and more RAM while the program is running, such as when an infinite loop occurs. Other common runtime errors include **divide-by-zero errors**, opening non-existent files for reading, calling functions that do not exist, or validation errors that have not been handled within the source code.

Runtime errors are often found during the testing stage of producing a software solution, but software can, and does, get released with the possibility of runtime errors still occurring. This is because these errors tend to occur only when certain environmental or technical conditions are met, or when unexpected inputs are entered into the software. In many cases, once a runtime error is found in a distributed piece of software, the software company releases **patches** and software updates after correcting the error.

Logic/semantic errors

Put simply, logic errors, also known as semantic errors, occur when the logic of a software program fails. This means that the source code is syntactically correct but the software solution, once compiled or interpreted, does not produce the expected output when run. In this case, the output is often unintended, undesirable or incorrect. For example, a function written to return the square root of a number may instead return the square of a number. Similarly, a function in an air-conditioning unit that checks to see if the temperature in a room is greater than 30 degrees before turning itself on may activate the air conditioning at 29 degrees instead.

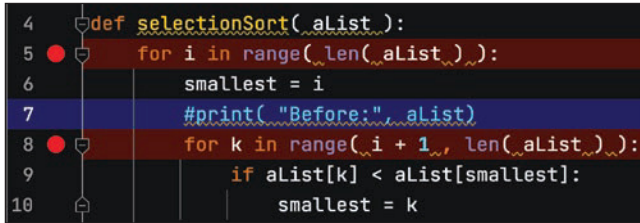
Logic errors can be very difficult to find, as there is nothing within a compiler or **interpreter** that will tell a programmer on which line a logic error appears. It is often up to the programmer as a human to construct test data and trace tables in order to track down the error manually.

Debugging techniques

Breakpoints

Breakpoints are markers set by a developer in a code editor or debugger that pauses the execution of the program during debugging. When the program execution reaches a line of code where a breakpoint is set, it stops, allowing the developer to examine the current state of the program.

In Figure 4.23, two breakpoints have been set; these are indicated in the IDE as red dots next to lines 5 and 8. When the code executes, the program will pause every time it runs those lines and display the contents of the current variables to the developer, as seen in Figure 4.24.

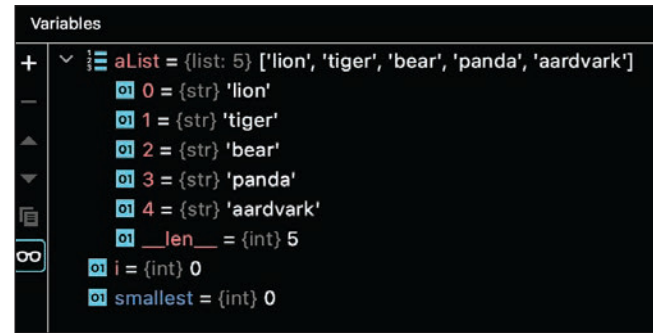


```

4 def selectionSort(aList):
5     for i in range(len(aList)):
6         smallest = i
7         #print("Before:", aList)
8         for k in range(i + 1, len(aList)):
9             if aList[k] < aList[smallest]:
10                smallest = k

```

FIGURE 4.23 Breakpoints in Python code, using the PyCharm IDE



Variables

- `aList = {list: 5} ['lion', 'tiger', 'bear', 'panda', 'aardvark']`
- `0 = {str} 'lion'`
- `1 = {str} 'tiger'`
- `2 = {str} 'bear'`
- `3 = {str} 'panda'`
- `4 = {str} 'aardvark'`
- `__len__ = {int} 5`
- `i = {int} 0`
- `smallest = {int} 0`

FIGURE 4.24 The output displayed when a breakpoint is reached when using the PyCharm IDE

Breakpoints are an essential tool in the software development process, particularly in the debugging phase. They help developers identify and fix issues in their code by allowing them to pause execution and inspect the state of the program at specific points.

Debugging statements

Debugging statements are lines of code added to a program specifically for the purpose of debugging. These often include print statements (such as `console.log` in JavaScript, `print` in Python or `System.out.println` in Java) that output the value of variables, the flow of execution, or specific messages to the console or log files.

Debugging statements provide insights into how data is being manipulated as the program runs. By observing the outputs of these statements, a developer can trace the execution flow and spot where things might be going wrong. These statements are easy to implement and, unlike breakpoints, do not require specialised IDE tools. They are useful for quick checks and simple debugging scenarios.

There are some disadvantages to using debugging statements. They can clutter the code if not removed after debugging and they also require a program to run in its entirety, unlike breakpoints, which might not be efficient for large or complex applications.

Test data and test tables

In order to systematically test that a module works, appropriate test cases must be written. A test case is a set of steps that a tester uses to determine if the element being tested works correctly. It involves selecting test data, writing testing procedures and determining expected results. It is particularly important that appropriate test data is selected so that test cases can be run. At a minimum, the test data selected must ensure full coverage of the algorithm when test cases are performed. This means that all paths of all control structures are tested fully.

There are four main types of test data. The first involves validation test data that tests the validation techniques that have been included in the module. Data for this purpose must be selected to test any instances of existence checking, type checking and range checking that have been included in the source code. At a minimum, this should involve selecting test data to check for both valid inputs and invalid inputs. For example, if checking for a valid age integer being input by a user, a valid input would be a positive integer, likely in the range of 0 to 120. An invalid input would be a negative integer, such as `-1`.

The second type of test data will test all conditional statements, including the operators within those conditions. Selecting appropriate data to do this may involve constructing truth

You can see a commented out debugging statement on line 7 of the sample code in Figure 4.23.

tables. It may also involve selecting appropriate **boundary values** to ensure full coverage of the operators included in the conditionals.

The third type of test data is designed to test all iterations, including any operators within those iterations. This may involve selecting or creating data that will ensure all conditions of those iterations are met. For example, if a WHILE loop is designed to iterate over each line of a file, test data would involve testing the following:

- when the file does not exist
- when the file exists but is empty
- when the file exists and has one line to read
- when the file exists and has more than one line to read.

The fourth type of test data involves creating test cases that will attempt to cover all of the functionality within the system from the perspective of a user. This typically means testing the graphical user interface to ensure that each screen within the application can be accessed appropriately and correctly. Typically, test cases are written based on design documents such as mock-ups, storyboards and/or site maps to ensure that the order and sequence of user interface elements are correct. This type of testing can be the most time-consuming, particularly if the application allows a user to take multiple paths to access particular screens. As a result, developers often write formal test cases that allow for automated software testing programs to be used to run these tests so they do not need to be performed manually. These test cases can be written to simulate the actions of a user navigating through the software solution. Within VCE Applied Computing, there is no formal requirement for students to develop, utilise or carry out automated testing of their modules.

Boundary values

Testing for boundary values involves selecting test data that will test the ‘boundaries’ of any condition or iteration within the code – that is, the maximum and minimum values available for a given input. Boundary values are particularly relevant for any algorithm that uses a range check. The general rule for boundary testing is that, at each boundary, test data should be selected to test both outside the boundary and inside the boundary.

For example, imagine a software solution designed for a Scout group that would only allow users to join if they are in the ‘Venturer’ age group (15–17 years). The boundary values for testing this age group can be determined by considering the type of data that is being collected. As this example involves testing for age, valid integers begin at 0 and are unlikely to go beyond 120. To test for the age of a particular age group would therefore require testing an age that is below the age group, within the age group, and above the age group.

The ages that are in range can be represented on a continuum:

0	1	2	3	...	14	15	16	17	18	19	...
						↑	↑	↑			

There are two boundaries for this particular test. One boundary exists at age 15 and the second boundary exists at age 17. The pseudocode that demonstrates these can be seen in Figure 4.25.

```

ALGORITHM checkVenturer( age )
{ Purpose: to check that a user is a Venturer based on age }
{ Inputs: age, as an integer }
{ Output: Boolean, True if age is valid, False if not }
BEGIN
    IF age > 14 AND age < 18 THEN
        RETURN True
    ELSE
        RETURN False
    ENDIF
END

```

FIGURE 4.25

Pseudocode to check an age range

In order to fully test this algorithm, the test data that must be selected is:

Age	Reason	Expected result
14	One year below the lower boundary of the valid range	False
15	The lower boundary of the valid range	True
17	The upper boundary of the valid range	True
18	One year above the upper boundary of the valid range	False

The age range in Figure 4.25 contains a two-sided boundary, but when you are testing a one-sided boundary, the most suitable values to check are: one below the boundary, the boundary itself, and one above the boundary.

When selecting above and below a boundary, test data should be in the smallest increment possible given the context. In this case it is one year but, for example, if an algorithm were to test for a price range, the smallest increment would be 0.01. Similarly, if testing an algorithm that uses boundaries involving hours or minutes in a day, the smallest increment would typically be one minute, thus boundaries at the hour would be 59, 0 and 1.

```

ALGORITHM abstractBoundaries()
BEGIN
    INPUT a, b
    IF a < b THEN
        PRINT "Condition met."
    ELSE
        PRINT "Condition not met."
    ENDIF
END

```

FIGURE 4.26 Abstract set of boundary conditions

For a more abstract example, the pseudocode in Figure 4.26 would require the following boundary values to be tested:

a	b	Reason	Expected result
a	a + 1	a is guaranteed to be less than b	Condition met
a	a	a is guaranteed to not be less than b, as it is equal	Condition not met
a	a - 1	a is guaranteed to not be less than b, as b will always be less than a	Condition not met

Test data: Library study room booking system

A library offers study rooms that can be booked by patrons. To manage the bookings efficiently, the library has implemented a booking system that only allows reservations for the current day. The library is open from 9 a.m. to 5 p.m., and the study rooms can be reserved from 3 p.m. onwards. If someone attempts to reserve a study room before 3 p.m., they will have their booking rejected.

An appropriate testing table for this scenario is shown in Table 4.11.

TABLE 4.11 Testing table for library study room booking system

Time (24-hour format)	Test description	Expected output
14:59	Time is one minute before 3 p.m.	Booking rejected
15:00	Time is 3 p.m. exactly	Booking accepted
15:01	Time is one minute after 3 p.m.	Booking accepted

For more complex algorithms, more test data needs to be selected, and this is when it can be very useful to construct a truth table to determine what that test data should be.

Consider the pseudocode example testing for multiple conditions in Figure 4.27.

```

ALGORITHM multiConditions()
BEGIN
    INPUT a, b, c, d
    IF a < b AND ( b < c OR c < d ) THEN
        PRINT "Condition met."
    ELSE
        PRINT "Condition not met."
    ENDIF
END

```

Truth tables are not in the Applied Computing study design, but they are incredibly helpful when determining the number of tests that need to be included to test all paths of execution.

FIGURE 4.27 Complex conditional with more than one logical expression

Consider also the truth table constructed for this algorithm, shown in Table 4.12.

TABLE 4.12 Truth table for Figure 4.27

a < b	b < c	c < d	b < c OR c < d	a < b AND (b < c OR c < d)
True	True	True	True	True
True	True	False	True	True
True	False	True	True	True
True	False	False	False	False
False	True	True	True	False
False	True	False	True	False
False	False	True	True	False
False	False	False	False	False

The test data required to fully test the conditions in Figure 4.27 must involve testing the conditions shown in each line of the truth table. For example, the second line of the truth table requires that the condition $c < d$ is not met, so a boundary test must be performed where c is not less than d (i.e. $c = d$).

For this pseudocode, at least eight test data elements must be written to test the conditional statement fully.

It can be useful to use an algebraic expression to map out which values are needed for a test condition to be met. For example, using the example again from Figure 4.27 and Table 4.12, the first line of the truth table can be interpreted algebraically in this way:

$a < b$	$b < c$	$c < d$	$b < c \text{ OR } c < d$	$a < b \text{ AND } (b < c \text{ OR } c < d)$
$a = b$	$b = c$	$c = d$	True	True

This is because each condition can be guaranteed to be True if the algebraic conditions are met.

The algebraic expressions can be further simplified:

$a < b$	$b < c$	$c < d$	$b < c \text{ OR } c < d$	$a < b \text{ AND } (b < c \text{ OR } c < d)$
$a = b$	$b = d - 1$	$c = d$	True	True

And finally:

$a < b$	$b < c$	$c < d$	$b < c \text{ OR } c < d$	$a < b \text{ AND } (b < c \text{ OR } c < d)$
$a = d - 1$	$b = d - 1$	$c = d$	True	True

Therefore, only a value for d needs to be chosen, as the values for a , b and c can all be determined from the value for d .

It is important that test data is selected systematically, rather than in an ad hoc manner, as otherwise it cannot be guaranteed that the source code will be logically correct throughout all of its algorithms. As a result, a significantly large set of test data may be required to test the system, and this is why many large software development companies use automated tools to construct and run tests on source code.

Trace tables

To prevent logic errors occurring, programmers often construct trace tables to validate the logic of the algorithms used in their source code. Trace tables simulate the execution of a program, referred to as the **flow of execution**. Given test data, each processing feature within an algorithm is executed, step by step, and, based on the test data, the values of the variables that change within that algorithm are tracked to ensure that the logic within the algorithm is correct. This systematic method of tracking the execution of code allows for the thorough testing of even the most complex of algorithms.

As an example, consider the pseudocode in Figure 4.28. This simplistic algorithm uses a WHILE loop control structure to print values until a condition is met. The trace table to represent the flow of execution of this pseudocode is in Table 4.13.

```

ALGORITHM sampleWhileLoop()
BEGIN
     $x \leftarrow 0$ 
     $y \leftarrow 0$ 
    WHILE  $x < 32$  DO
         $x \leftarrow x + 8$ 
         $y \leftarrow y + 4$ 
    ENDWHILE
    PRINT  $x$ ,  $y$ 
END

```

While trace tables are not in the Applied Computing study design, they are very useful to detect logic errors in code.

FIGURE 4.28

Pseudocode example of a WHILE loop

TABLE 4.13 Trace table for the pseudocode in Figure 4.28

Step	Statement	x	y	output
1	$x \leftarrow 0$	0		
2	$y \leftarrow 0$	0	0	
3	While $x < 32$ Do	0	0	
4	$x \leftarrow x + 8$	8	0	
5	$y \leftarrow y + 4$	8	4	
6	While $x < 32$ Do	8	4	
7	$x \leftarrow x + 8$	16	4	
8	$y \leftarrow y + 4$	16	8	
9	While $x < 32$ Do	16	8	
10	$x \leftarrow x + 8$	24	8	
11	$y \leftarrow y + 4$	24	12	
12	While $x < 32$ Do	24	12	
13	$x \leftarrow x + 8$	32	12	
14	$y \leftarrow y + 4$	32	16	
15	While $x < 32$ Do	32	16	
16	EndWhile	32	16	
17	Print x, y	32	16	32, 16

Another example, where a logic error exists within an algorithm, can be seen with the algorithm in Figure 4.29.

As this algorithm requires input from a user, the trace table can only be constructed using test data. For the purposes of this example, the test data is as follows (Table 4.14) and the flow of execution is tested twice, with two sets of inputs and the expected discounted price being listed with the test data.

```

ALGORITHM applyDiscount()
BEGIN
    INPUT fullPrice
    INPUT discPerc

    discPrice  $\leftarrow$  fullPrice - discPerc
    RETURN discPrice
END

```

FIGURE 4.29
Pseudocode containing a logic error**TABLE 4.14** Test data for the pseudocode in Figure 4.29

fullPrice	discPerc	Expected result
20.00	5	19.00
50.00	50	25.00

TABLE 4.15 Trace tables for the pseudocode in Figure 4.29

Step	Statement	fullPrice	discPerc	discPrice	output
1	Input fullPrice	20			
2	Input discPerc	20	5		
3	discPrice ← fullPrice – discPerc	20	5	15.00	
4	Return discPrice	20	5	15.00	15.00

Step	Statement	fullPrice	discPerc	discPrice	output
1	Input fullPrice	50			
2	Input discPerc	50	50		
3	discPrice ← fullPrice – discPerc	50	50	0.00	
4	Return discPrice	50	50	0.00	0.00

After completing the trace tables, the test data can be completed and the results are as shown in Table 4.16.

TABLE 4.16 Completed test data for the pseudocode in Figure 4.29

fullPrice	discPerc	Expected result	Actual result
20.00	5	19.00	15.00
50.00	50	25.00	0.00

From viewing the actual results from the trace table and the test data, it is clear that the algorithm is not calculating a correct discounted price given the inputs it is receiving; it is merely subtracting the discount percentage as if it were a dollar value to be discounted. The corrected algorithm can be seen in Figure 4.30.

```

ALGORITHM applyDiscount ()
BEGIN
    INPUT fullPrice
    INPUT discPerc

    discAmount ← fullPrice * (discPerc / 100)
    discPrice ← fullPrice – discAmount
    RETURN discPrice
END

```

FIGURE 4.30 Discount pseudocode after the logic error is fixed

Traditionally, a programmer would produce trace tables manually to test code and check for logic errors. Many programming languages today, however, have **integrated development environments (IDEs)** that allow a programmer to trace the flow of execution through a debugger built into the IDE.

4.5 THINK ABOUT APPLIED COMPUTING

Look up the documentation for the IDE that you use with your selected programming language. If it supports tracing the flow of execution, run the tracing function over some of your code to see how it works.

4

CHAPTER SUMMARY

Essential terms

abstraction allows software developers to create models that represent real-world objects or systems in a simplified manner, enabling easier understanding and management of software

alternative execution code that is run if a condition is not met

arguments specific inputs passed into a function that act as local, temporary variables

arithmetic operator add, subtract, multiply, divide, raise to a power

Boolean a logical data type; Boolean data can hold one of only two possible values, true or false

boundary values maximum and minimum edge values possible for a given input

breakpoint a marker set by a programmer within the source code of a program to pause its execution at a specific point during debugging

built-in functions functions that the creators of the programming language have written to execute common sequences of code

camel case a naming convention where each word is capitalised except the first, with no spaces or underscores (e.g. camelCase)

chained selection a conditional statement that handles more than one possible conditional outcome

compiler a program that turns source code into machine language that can be executed by a computer processor

conditional operator used to evaluate a condition and return a value based on the result of that condition

debugging the process of identifying and removing errors from computer software

debugging statement a piece of code inserted into a program by a programmer to facilitate debugging and troubleshooting

divide-by-zero errors runtime errors that occur when a number is divided by zero, which is mathematically undefined

DO/WHILE loop an iteration over a set of instructions, conditions and/or iterations that is repeated for as long as a condition is met; it is always run at least once

encapsulation hides the internal state of an object from the outside world and only exposes the necessary functionalities through well-defined interfaces; this supports modularity, maintainability and information hiding

event a special type of method that is called when an object's state changes

existence check a validation process performed to determine whether a variable, file, database record or other resource is present and accessible before attempting to use or manipulate it

expected results output expected from an algorithm, assuming it is logically correct

flow of execution the order in which instructions, conditions and iterations are executed or evaluated

FOR loop an iteration over a set of instructions that is repeated a set number of times

function a sequence of related code that has been given a name that can be called from other points in the source code

function call the use of code within an algorithm to execute the contents of a function

function declaration a sequence of code that names a function and its arguments

function definition to define (write) the contents of a function

function visibility the scope within which a function can be accessed or called in a program

global variables variables declared outside any function or block, accessible throughout the entire program

header comment a set of meaningful comments at the top of a source code file, outlining information such as the name of the file, its purpose, the author's name and the date of creation

Hungarian notation a naming convention where the name of a variable or function includes a prefix that indicates its type or purpose

infinite loop an iteration that will never reach the condition upon which it can terminate

instruction a unit of code that can be executed by a compiler or an interpreter

integrated development environment (IDE) software that provides tools to programmers to aid in programming, such as source code editing, syntax highlighting, code completion, debugging aids, or tools to help construct a user interface

internal documentation notes and code comments contained within source code that describe the code

interpreter a computer program that directly executes source code without needing to have it compiled beforehand

iteration also known as a loop; the repetition of a section of code until a condition is met

local variables variables declared within a function or block, accessible only within that function or block

logic error when source code is syntactically correct but contains an error resulting in unintended, undesirable or incorrect output

logical operator a Boolean operator used to combine expressions, such as AND, OR

memory leak the failure of a program to release memory that is no longer needed, causing impaired performance, application failure and/or system failure

method an action an object can carry out – for example, window.refresh, golfClub.swing

naming convention a set of rules for choosing the character sequence to be used for identifiers in code such as variables and function names

nested selection statement when a condition contains one or more additional conditions within its structure

object-oriented programming (OOP) software design is organised around objects, which encapsulate data and behaviour, which support modularity, reuse and flexibility in software development

parameters see **arguments**

pass by reference to pass data into a function as an argument so that it can be modified without needing to be returned

pass by value to pass data into a function as an argument so that it cannot be modified without needing to be returned

patches updates to software that fix bugs, improve functionality or enhance security without a full version release

pointer a programming language element that stores the memory address of another data value located in memory; the pointer 'points' to that memory space

procedural programming language programs are organised as sequences of instructions that operate on data; other programming types are: functional, recursive

procedure also known as method, function or subroutine; a named block of code that performs a specific task or operation within a program

range check a validation process used to ensure that a value falls within a specified range of acceptable values

REPEAT/UNTIL loop an iteration over a set of instructions that is repeated for as long as a condition is not met; it will always execute at least once

return value a value or set of values that is passed back to the origin of a calling function, often to be assigned to a variable, used in an equation, or tested within a conditional statement

runtime error an error that occurs while a program is running

selection statement a control structure that allows a programmer to write lines of code that are only run when a particular requirement is met; also referred to as a selection

sequence a set of instructions that executes line by line in the order in which it is written

snake case a naming convention where each word is separated by an underscore, and all letters are usually in lower case (e.g. snake_case)

subroutine a named block of code within a program that performs a specific task or operation; subroutines are reusable units of code that can be called from different parts of a program to perform the same task

syntax error an error, often typographical, in source code that violates the set of rules that define a programming language

test case a set of steps that a tester uses to determine if the element being tested works correctly, often outlining test data, testing procedures and expected results

test data data that has been specifically identified to be used in a test case

trace table a table used to test an algorithm, typically by hand, to ensure that no logic errors occur

truth table a table used to represent all of the combinations of values for inputs and their outputs, typically used to test conditional statements

type check a validation process performed to verify that a variable or expression has a specific data type or conforms to a particular data format constraint

variable a method of storing and labelling data to be referenced and manipulated in a computer program

version control system a tool that helps manage changes to source code over time, keeping track of modifications and allowing collaboration

WHILE loop an iteration over a set of instructions that is repeated for as long as a condition is met

Important facts

- 1 Variables are used to store data so they can be retrieved and changed in a program.
- 2 Instructions are a unit of code that can be executed by a compiler or interpreter.
- 3 Definitions are a type of instruction that assigns values to variables.
- 4 Statements are single lines of code that, when executed, perform a single action.
- 5 There are three fundamental control structures in programming: sequences, conditions and iterations.
- 6 Sequences are sets of instructions that execute line by line; every line of code is run in the order in which it is written.
- 7 Selection statements are control structures that allow programmers to write lines of code that are only run when a particular conditional requirement is met.
- 8 Code within a conditional statement can contain instructions, sequences, other conditional statements or iterations.
- 9 Selection statements are Boolean, as they are run based on the result of a condition being evaluated as either True or False.
- 10 Alternative executions are run when a condition is not met; they are typically run by using an 'else' statement.
- 11 Selection statements can be written to include one or more logical expressions; the number of logical expressions in a conditional statement can in theory be infinite.
- 12 Truth tables are used to test each combination of values within a condition.
- 13 Truth tables can be used to help determine if logical errors exist.
- 14 Chained selection statements are used when a condition needs to be tested more than once; switch/case conditionals also allow this.
- 15 Nested selection statements occur when one condition is placed inside another condition; there can in theory be an infinite number of nested conditionals.
- 16 Iterations repeat sections of code multiple times until a condition is met.
- 17 There are four types of iterations: WHILE, DO/WHILE, FOR and REPEAT/UNTIL.
- 18 WHILE loops run for as long as a condition is met; they are used when it is unknown how many times the loop will execute.
- 19 DO/WHILE loops are like WHILE loops but will always run at least once; WHILE loops may not necessarily run if the condition is never met in the first place.

- 20** FOR loops are run a predefined number of times.
- 21** REPEAT/UNTIL loops will run for as long as a condition is not met; they will stop only when a condition is met.
- 22** A function is a sequence of related code to which a programmer has given a name.
- 23** When called, code within a function typically returns a value.
- 24** Function declarations name a function and its arguments.
- 25** Built-in functions can be used without needing to define the function yourself; they are provided by the creators of the programming language.
- 26** Functions can require arguments, also known as parameters, which are temporary variables that are part of the function declaration.
- 27** Local variables exist only within a function, whereas global variables exist throughout the whole program.
- 28** Classes allow programmers to group together related functions and variables in one place, creating a template for objects.
- 29** Methods are functions that exist within classes; one type of method is an event, which is a method that is called when an object's state changes.
- 30** Internal documentation exists within source code and includes the notes and comments a programmer makes about the program.
- 31** Internal documentation has no impact on the compilation or execution of software; it is ignored by the compiler or interpreter.
- 32** Internal documentation should include header comments, documentation of classes and methods, function comments, and comments describing complex portions of algorithms within the source code.
- 33** Internal documentation should be meaningful and non-trivial; it should not simply state or restate what is occurring in the code.
- 34** Testing is one of the most important steps in building a software program.
- 35** Appropriate testing activities include determining complete sets of test data, stating the expected results, conducting tests, recording results and then correcting any errors.
- 36** Debugging is a testing method that includes finding errors through creating trace tables and testing the system using test data.
- 37** Three main types of errors can occur in software solutions: syntax errors, runtime errors and logic errors.
- 38** Syntax errors occur when source code has not followed the predefined language rules of the chosen language; these errors prevent a solution from compiling.
- 39** Runtime errors occur when source code has been compiled and is being executed; the error causes the program to crash or print error messages.
- 40** Logic, or semantic, errors occur when the logic of a software program fails; in this case, while the code is syntactically correct, it does not produce the output that is expected.
- 41** Logic errors are best detected through the use of test data and trace tables.
- 42** There are four main types of test data: validation test data, selection testing data, iteration testing data and test cases covering all functionality from the perspective of the user.
- 43** Selection and iteration testing data involves testing for boundary values of any and all conditions and iterations within the code.
- 44** The general rule for boundary testing is that at each boundary, test data should be selected to test outside the boundary as well as inside the boundary.
- 45** Truth tables are useful to help determine test data as they show all of the potential outcomes of logical expressions used in conditions and iterations.
- 46** Trace tables simulate the flow of execution of source code.



Processing features of a programming language

- 1 Explain how a variable differs from an instruction.
- 2 Which of the following are most likely to be variables?
 - a seekAlternateRoute()
 - b False
 - c potato
 - d user.firstName
 - e 2019
 - f isValid
 - g "chocolate"
- 3 Categorise the following as definitions or statements.
 - a `intNumCustomers ← 7`
 - b `executePrintJob()`
 - c `strCustomerID ← getCustomerID()`
- 4 Name three types of control structures.
- 5 How does a sequence differ from an iteration?
- 6 Pirru writes some code to loop over a set of data. Inside the loop she has a print statement that will print a number to the screen every time the source code within the loop is run. She runs the code, and it runs without errors, but nothing is printed to the screen. What are the possible iterations Pirru could have used? Why?
- 7 State the values that will be returned or displayed in the following examples of pseudocode.

a

```
x ← 1
y ← 3
z ← 7
IF (x < z) OR (y > z) THEN
    RETURN True
ELSE
    RETURN False
ENDIF
```

b

```
yearsService ← 10
IF yearsService < 10 THEN
    RETURN "Newbie"
ELSEIF yearsService < 20 THEN
    RETURN "Experienced"
ELSE
    RETURN "Expert"
ENDIF
```

- 8 How are functions represented in pseudocode?
- 9 Outline the difference between a function declaration and a function definition.
- 10 How is a class different from an object?



TEST YOUR KNOWLEDGE

11 What is the relationship between a method and an event?

12 Show how methods are represented in pseudocode.

Internal documentation

13 Justify the use of internal documentation.

14 Outline the essential elements that should be included in a header comment.

15 What special characters are used to indicate comments in the programming language you are learning?

Testing and debugging techniques

16 Describe the difference between a test case and test data.

17 Explain the difference between a syntax error and a runtime error.

18 What is the purpose of a truth table in relation to testing?

19 Explain what a boundary value is.

20 Why is it important to use test data that focuses on boundary values?

21 Provide two reasons why trace tables are useful when debugging.

22 Complete a trace table for the following algorithm:

```
:  
: ALGORITHM computeResult()  
: BEGIN  
:     x ← 7  
:     y ← 1  
:  
:     WHILE y < x DO  
:         y ← x * y  
:         x ← x + x  
:     ENDWHILE  
:     PRINT x, y  
: END
```



Major programming task: Hangman

Your task is to design, develop and evaluate a program that will allow a user to play the game 'Hangman'. The program will randomly select a word from a list that the user will then attempt to guess by suggesting letters from the alphabet. Each time a letter is guessed incorrectly, an element of a hanged man is drawn. If all elements of the hanged man are drawn, the player loses. If the player guesses all of the letters in the word, or guesses the word itself, before the hanged man is drawn, the player wins. A player can guess the word at any time, but if they guess incorrectly, an element of the hanged man is drawn. If you haven't played Hangman before, you can find out the rules at the weblink. There are many online Hangman games you can search for and play as part of your research to create your own version of the game.

- 1 Describe the scope, constraints and functional requirements of the program.
- 2 Design the program's interface with a mock-up.
- 3 Write pseudocode to describe the processing strategies involved in developing the program.
- 4 Develop the solution, using good naming conventions and internal documentation.
- 5 Construct a set of test data that will test all conditions – for example, when a letter is found, when more than one letter is found, and when no letter is found.
- 6 Test the solution and fix all bugs using debugging statements and breakpoints.



Weblink
Hangman

PREPARING FOR

Unit

1

OUTCOME 2

On completion of this unit the student should be able to interpret teacher-provided solution requirements to design and develop a software solution using an object-oriented programming language.

Steps to follow

To achieve this outcome, you will draw on key knowledge and key skills outlined in Area of Study 2. This outcome requires that you use a programming language to create a working solution to a teacher-provided design brief, undertaking the problem-solving activities of design, coding, validating, testing and creating internal documentation.

Ensure that you read the software requirements your teacher gives you carefully, paying close attention to scope, constraints, functional requirements and non-functional requirements.

Documents required for assessment

- Design documents: data dictionary, mock-ups, pseudocode
- Source code and all related files (e.g. images, data files)
- Internal documentation within the source code
- Evidence of testing using appropriate test data
- All documents used to construct your tests, including test data, any truth tables used to select your test data, and trace tables; these should be kept and submitted

Assessment

You will be assessed on the following measures:

- the accuracy and completeness of your design documents
- your choice of data types and data structures
- appropriateness of processing features selected, created and used
- use of programming naming conventions
- validation techniques
- internal documentation
- debugging techniques
- thoroughness of testing.



Unit 1 exam practice questions

For Unit 1, you need to ensure that you have a sound understanding of the required key knowledge as well as the ability to demonstrate the key skills. When revising, it is useful to relate the key knowledge to real-world applications to enhance comprehension. Real-world applications provide concrete examples of abstract concepts, making them more tangible and easier to understand. For example, understanding keyword detection and pattern matching in the context of developing a simple ELIZA chatbot can illustrate how these concepts are applied in natural language processing.

You need to be familiar with the design tools that can be used to develop both data visualisations and software solutions. There are key concepts that overlap both outcomes, and you should revise these carefully. For example, data types and structures are relevant to both data visualisations and software solutions.

Thoroughly read and understand all questions, including the case study when provided. Underline command terms and key phrases in the question to help you understand the intention of the question, especially for theory and short-answer sections.

For multiple-choice questions, learn to use the process of elimination in questions where you are unsure of the answer. Often, you can narrow down the choices to two, increasing your chances of selecting the correct answer.

Short-answer questions require clear and concise responses. Focus on directly answering the question, avoiding unnecessary details. Demonstrate your knowledge by using appropriate technical terms. However, ensure that you fully understand the terms you use.

Practise managing your time effectively. Allocate time to each section based on its weight in the overall grade and your proficiency in that area. If time permits, review your answers, focusing on sections you found challenging. This can help catch and correct any mistakes or oversights.

Multiple-choice questions

Question 1 (1 mark) KK1.1.1

Which of the following is an example of quantitative data?

- A. Names of cities in a country
- B. Types of fruits sold in a market
- C. Gender of participants in a study
- D. Rating movies on a scale from 1 to 5

Question 2 (1 mark) KK1.1.3

Factors affecting the quality of data and information are:

- A. accuracy, bias, context, relevance.
- B. accessibility, clarity, size, structure.
- C. bias, integrity, relevance, reliability.
- D. clarity, context, relevance, structure.



EXAM PRACTICE

Question 3 (1 mark) KK1.1.5

The *Privacy Act 1988* applies to companies with over \$3 million turnover and requires those companies to:

- A. keep computer systems up to date.
- B. keep information for its stated purpose only.
- C. seek permission to use individuals' photos and images.
- D. notify individuals when their data is stolen by hackers or insiders.

Question 4 (1 mark) KK1.2.2

What characteristic distinguishes the Boolean data type from others?

- A. It can represent dates and times.
- B. It can only hold two possible values.
- C. It can store alphanumeric characters.
- D. It can store both integers and floating-point numbers.

Question 5 (1 mark) KK1.2.12

The primary function of breakpoints in the debugging process is to:

- A. generate error reports.
- B. display relevant test data.
- C. compare expected and actual output.
- D. stop program execution at a specific point for examination.

Short-answer questions: Data analysis

Sample case study: Data analysis

The governing board of a company decides that a company-wide survey will be conducted to determine the level of employee satisfaction with the operation within the workplace. The intention is to gather insights and perspectives on the operation of the business.

Question 1 (2 marks) KK1.1.1

Describe a strategy that would allow the collection of relevant data and information from employees.

Question 2 (2 marks) KK1.1.6

Identify any ethical issues that may arise from collecting information from employee interviews.

Question 3 (4 marks) KK1.1.3, KK1.1.4

After analysing and aggregating the data, a data visualisation that arranges the findings is prepared. Identify and describe two design factors that should be considered when choosing how to present the data and information in the data visualisation.

**Question 4** (2 marks) *KK1.1.9*

When preparing the data visualisation, several design tools could be used. Identify a design tool and describe how that design tool would assist the development of a data visualisation.

Question 5 (3 marks) *KK1.1.12, KK1.1.13, KK1.1.14, KK1.1.15*

Given the amount of data being collected, and the purpose of the data collection and data visualisation being created, which software tool would be most appropriate to use to manipulate the data: spreadsheet software or database software? Justify your selection, referring to elements of the case study.

Short-answer questions: Programming**Sample case study: Programming**

Following acceptance of a software requirements specification (SRS), the design team hands over to the development team to begin coding a solution. One of the team members is enthusiastic to try a new programming language; however, the team leader has cautioned that the new language must be compliant with object-oriented programming (OOP) principles.

Question 1 (3 marks) *KK1.2.7*

Explain one principle of object-oriented programming, using an example as part of your explanation.

Question 2 (2 marks) *KK1.2.9*

The development team has had problems maintaining a working master version of its software solution. Clear communication between team members is an important strategy in developing an efficient and effective solution. Describe the purpose of a naming convention and how that might assist clear communication.

Question 3 (2 marks) *KK1.2.10*

The development team has not used internal documentation in its software solution so far. List and describe two purposes of internal documentation.

Question 4 (2 marks) *KK1.2.4*

The software requirements specification contained information about the amount of money available for the project as well as the hardware that had already been purchased to assist in the creation of the software solution. Where would the development team have found this information in the SRS? Explain.



EXAM PRACTICE

Question 5 (4 marks) *KK1.2.5*

The software requirements specification contains the following functional requirement:
'The system will respond to user inputs based on keyword detection and pattern matching.' Create a flowchart that best represents this functional requirement.

Question 6 (2 marks) *KK1.2.1*

One of the team members wants to use AI-based code generators for the majority of the project. Explain how AI-based code generators might be used to assist in the coding of the solution.

Unit 2

INTRODUCTION

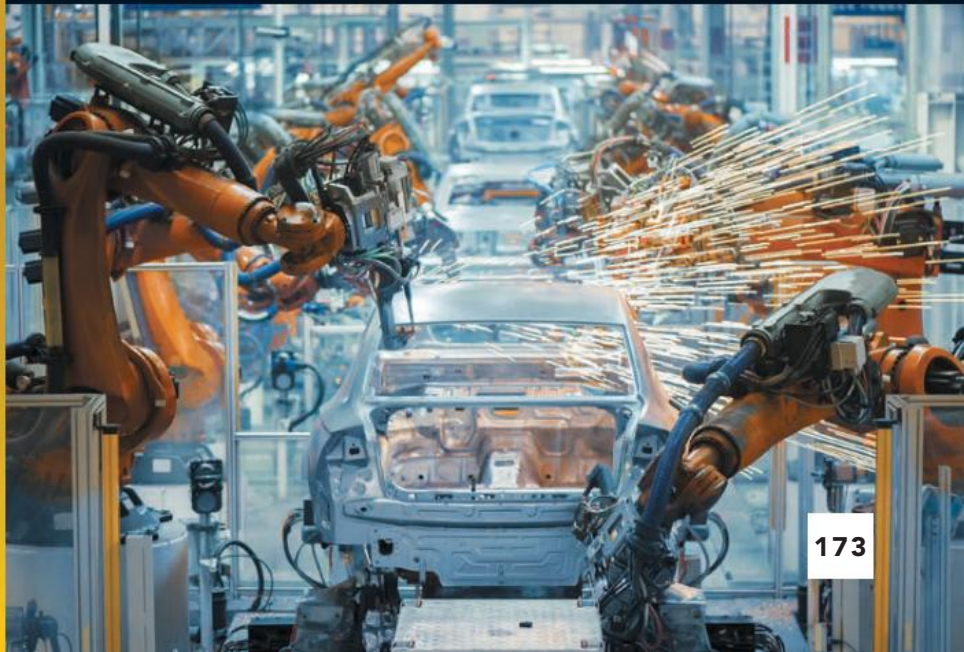
During Unit 2 of Applied Computing you will look at a range of emerging and innovative technologies, such as artificial intelligence, machine learning, assistive technologies, smart devices, drones and blockchain. You will gain an understanding of how networks are used to communicate data and information, identify threats to data and information in a networked environment and investigate a range of security measures that can be used to protect against these threats.

Area of Study 1 – Innovative solutions

OUTCOME 1 You will work collaboratively in groups to develop an innovative solution to an identified need or opportunity. Each group will follow the problem-solving methodology to create a solution to the need or opportunity using innovative or emerging technology. Each group will also prepare a progress plan to assist in the management of the task.

Area of Study 2 – Cyber security

OUTCOME 2 Your teacher will provide you with a case study. You will be required to examine the capabilities and vulnerabilities of a network, design a network solution, discuss threats to data and information and propose security strategies to protect the data and information.



KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

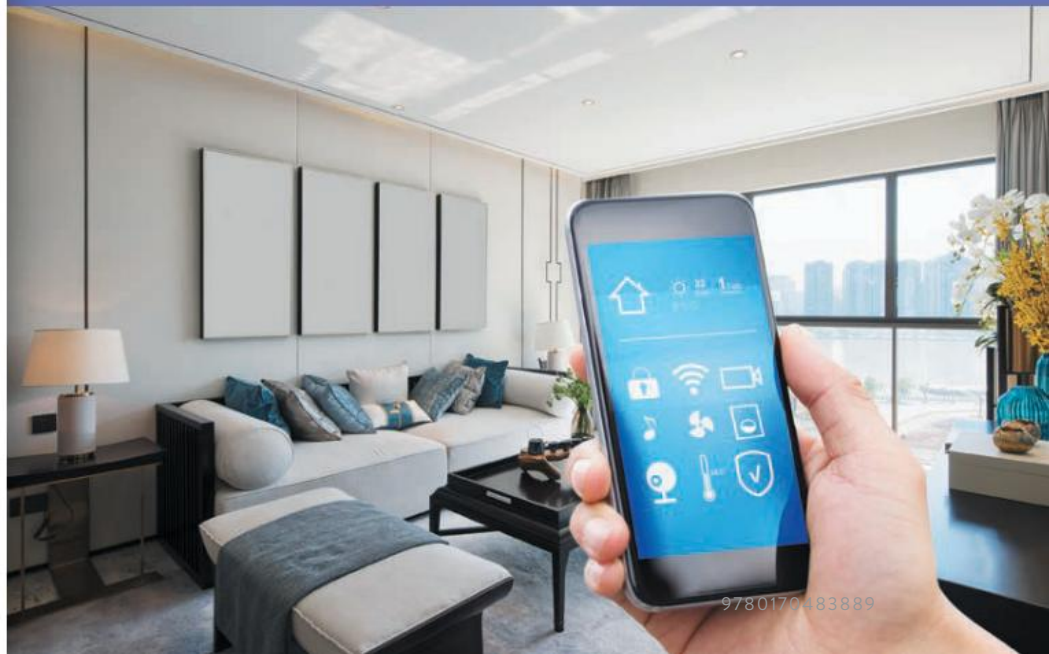
- KK2.1.1: the role of curiosity, ingenuity and the United Nations' Sustainable Development Goals to inspire and drive innovation
- KK2.1.2: functions and capabilities of digital systems used by individuals and organisations, such as: assistive technologies, financial services, global positioning system (GPS) devices, robotics, traffic management
- KK2.1.3: the impact of emerging technologies, such as: automation, cyberbullying, productivity, economic issues (deskilling, job loss, misuse), the decline of physical human interactions and interpersonal skills.

Victorian Curriculum and Assessment Authority, Victorian Certificate of Education Applied Computing Study Design 2025–2028, VCAA, 2024, p. 32

FOR THE STUDENT

This chapter, along with Chapter 6, covers the content for Unit 2, Area of Study 1, Innovative solutions. For this unit, you are required to work in groups to analyse, design, develop and evaluate an innovative solution to an identified need or opportunity involving a digital system. In this chapter, functions and capabilities of digital systems, the types of digital devices used and emerging trends in technology are introduced. A range of technology used in current and emerging technology is also introduced.

For Unit 2, Outcome 1, you are expected to work in groups to create a solution to an identified need or opportunity. The emphasis is on collaboration for this task as you apply problem-solving skills to your innovative solution designs. Chapter 6 deals with approaches to problem solving, and also covers knowledge required to complete Unit 2, Outcome 1.



Inspiring and driving innovation

Curiosity, ingenuity and the United Nations' Sustainable Development Goals (SDGs) help to inspire and drive innovation, particularly in fields that intersect with global challenges and aspirations for a sustainable future.

Curiosity and ingenuity

Curiosity is the intrinsic desire to learn, explore and understand the world around us. It is the starting point of all scientific inquiry and innovation, prompting individuals and communities to question the status quo and seek out new knowledge and solutions. Curiosity drives the initial spark that leads to the discovery of new concepts, the exploration of alternative methods and the questioning of existing solutions, laying the groundwork for innovative thinking and approaches.

Ingenuity, on the other hand, is the creative ability to solve problems and overcome challenges in novel ways. It involves applying knowledge, creativity and resourcefulness to devise effective solutions to complex problems. Ingenuity is essential in the innovation process, as it translates curiosity-driven questions and challenges into practical, impactful solutions. It requires a deep understanding of the problem at hand, as well as the ability to think laterally and combine ideas into cohesive, novel solutions.

Innovation using curiosity and ingenuity

Imagine a scenario where a team of software developers is tasked with creating an educational platform aimed at enhancing learning experiences for students with diverse needs. This platform is envisioned to be inclusive, engaging and adaptable to various learning styles and abilities.

Curiosity

Curiosity plays a foundational role at the outset of the project. The developers begin by asking questions and seeking to understand the diverse needs of students. They explore existing educational technologies, research teaching theories and consider the challenges faced by students with diverse needs. This inquisitive approach leads them to identify gaps in current offerings, such as a lack of personalised learning paths or inadequate support for students with specific learning requirements.

Their curiosity drives them to explore a wide range of disciplines, including cognitive psychology, user interface design and adaptive learning technologies. They question how these elements can be integrated into a single platform to create a more effective learning environment. Through this exploratory process, the team gathers the necessary insights and knowledge that will inform the design and functionality of the new software.

Ingenuity

After completing their curiosity-driven research, the team moves to the phase where ingenuity plays a role. They must now apply their creativity, technical skills and the understanding they have acquired to solve the complex problem at hand: designing an educational platform that is both effective and accessible to all students.

Ingenuity is evident in how the team devises innovative features for the platform, such as adaptive learning algorithms that adjust the difficulty level and content presentation based on individual student performance and preferences. They also creatively integrate multimedia elements and gamification to make learning more engaging for students with varying interests and attention spans.

**CASE
STUDY**





The team faces technical challenges, such as ensuring that the platform is accessible to students with visual or hearing impairments. Ingenuity shines through as they develop solutions such as text-to-speech functionality, customisable interface options and sign language avatars to make the platform truly inclusive.

How they compare

In this scenario, curiosity is the driving force behind the team's desire to understand and explore the complex needs of a diverse student population and the potential of technology to meet those needs. It is the spark that ignites the quest for knowledge and the deep dive into research that precedes the creation process.

Ingenuity, on the other hand, is the creative application of that knowledge and understanding. It involves the synthesis of information, the invention of novel solutions and the overcoming of technical hurdles to bring the concept of an inclusive educational platform to life.

While curiosity lays the groundwork by expanding the team's understanding and uncovering new possibilities, ingenuity is the catalyst that transforms these insights and ideas into a tangible, innovative software solution that has the potential to revolutionise learning for students with diverse needs.

United Nations' Sustainable Development Goals

The United Nations' Sustainable Development Goals (SDGs) provide a framework of 17 interlinked global goals designed to be a 'blueprint to achieve a better and more sustainable future for all' by 2030. These goals address the world's most pressing challenges, including poverty, inequality, climate change, environmental degradation, peace and justice. The SDGs serve as a powerful motivation for innovation by setting clear, ambitious targets for global development. They inspire individuals, organisations, governments, as well as the private sector, to innovate by providing a shared agenda that focuses efforts on key areas where innovation can have the most significant impact.

TABLE 5.1 The 17 Sustainable Development Goals (SDGs)

Goal number	Sustainable Development Goal
1	No poverty
2	Zero hunger
3	Good health and well-being
4	Quality education
5	Gender equality
6	Clean water and sanitation
7	Affordable and clean energy
8	Decent work and economic growth
9	Industry, innovation and infrastructure
10	Reduced inequalities
11	Sustainable cities and communities
12	Responsible consumption and production
13	Climate action
14	Life below water
15	Life on land
16	Peace, justice and strong institutions
17	Partnerships for the goals

Source: United Nations' Sustainable Development Goals. <https://sdgs.un.org/goals>

The SDGs, with their broad and interconnected objectives, encourage holistic thinking and innovation that not only addresses specific problems but also contributes to the larger goal of sustainable development. Innovations inspired by the SDGs are diverse, ranging from technological advancements in renewable energy and sustainable agriculture to new business models that promote social equity and environmental stewardship.

Software applications have become powerful tools in advancing the United Nations' Sustainable Development Goals by addressing complex global challenges through innovative solutions. The case study below provides some examples of how software applications contribute to meeting specific SDGs.

Read more about the 17 SDGs on the United Nations' Sustainable Development Goals website.



WebLink
United Nations'
Sustainable
Development
Goals

Applications meeting SDG goals

SDG 3: Good health and well-being – telehealth platforms

Telehealth platforms have revolutionised access to healthcare services, especially in remote and under-served areas. These applications allow patients to consult with healthcare professionals via video calls, messaging and remote monitoring services. This innovation improves access to medical advice, reduces the need for travel and ensures continuous care for chronic conditions, thereby contributing significantly to improving health and wellbeing.

SDG 6: Clean water and sanitation – water monitoring systems

Software applications that monitor water quality and usage help in managing water resources more efficiently. These systems can predict water demand, detect leaks and monitor contamination levels in water bodies. By providing real-time data and analytics, such applications support efforts to ensure the availability and sustainable management of water and sanitation for all.

SDG 13: Climate action – carbon footprint calculators

Software applications that calculate an individual's or organisation's carbon footprint help raise awareness about the impact of daily activities on the environment. These apps suggest ways to reduce emissions, such as by changing travel habits, energy consumption and dietary preferences, thereby encouraging actions that contribute to climate change mitigation.

SDG 15: Life on land – wildlife conservation apps

Software applications used in wildlife conservation efforts enable conservationists to collect data on wildlife sightings, poaching incidents and habitat changes. These tools support the protection of biodiversity by facilitating more effective conservation planning and action.

CASE STUDY



By fostering curiosity and ingenuity and aligning these qualities with the aspirational framework of the SDGs, we can inspire and drive innovation that is not only technologically advanced but also socially and environmentally responsible. This approach ensures that innovations contribute positively to global **sustainability**, equity and wellbeing, ultimately leading to a more sustainable, just and prosperous world for future generations.

Functions and capabilities of digital systems

There are many existing technologies that are designed to make the lives of individuals and organisations easier. These include:

- assistive technologies
- financial services
- global positioning system (GPS)
- robotics
- traffic management.

Assistive technologies

Assistive technology is any technology that allows an individual or organisation to complete a task that they would not normally be able to complete, or that makes it easier or safer to complete day-to-day activities.

Devices and situations in which assistive technology is used include:

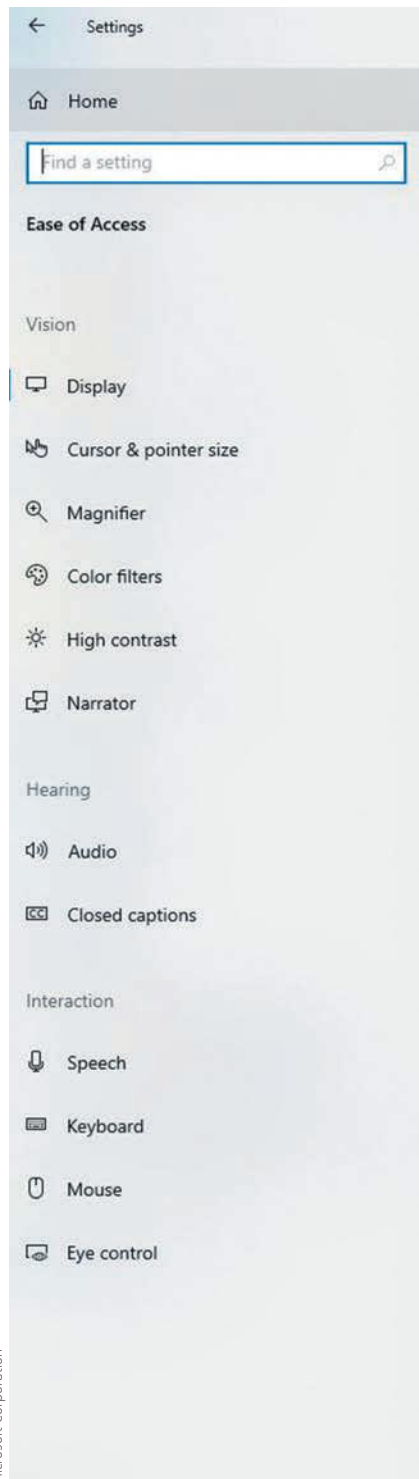
- accessibility aids
- cognitive aids
- personal emergency response systems
- home automation
- mobility aids.

Accessibility aids

Accessibility aids are designed to make everyday tasks accessible to everyone, especially those people who have an impairment or disability. Such aids include software designed to make it easier for someone to access data and information while using a digital or information system. This can include text-to-speech software, which allows users to hear information rather than read it on a screen and is particularly useful for people with vision problems. Speech-recognition software allows a user to input data without the need for a keyboard. This can be useful for people who have limited use of their hands or arms.

Other accessibility aids include:

- closed caption software, which allows hearing-impaired users to read, rather than listen to, the spoken words in audio and video files
- keyboard shortcuts and sticky keys for those users who prefer not, or are unable, to use a mouse
- visual cues to alert a user to a particular action, rather than sound alerts
- contrast control, text and icon resizing
- contrast control or alternative text for images to make the data and information easier to read.



Display

Make your display easier to see.

Make text bigger

Sample text

Drag the slider until the sample text is easy to read, then click Apply

A  A

Apply

Make everything bigger

Change the size of apps and text on the main display

100% (Recommended) ▼

[Change the size of apps and text on other displays](#)

[Change the size and color of your cursor and mouse pointer](#)

Make everything brighter

[Change brightness automatically or use night light](#)

Simplify and personalize Windows

Show animations in Windows

 On

Show transparency in Windows

 On

Automatically hide scroll bars in Windows

 On

Show notifications for

5 seconds ▼

Show desktop background image

 On

[Personalize your background and other colors](#)

FIGURE 5.1 Access settings in Windows

Cognitive aids

Cognitive aids assist people with a physical, cognitive or emotional impairment that affects processes such as attention, memory, navigation, planning or sequencing of events or tasks. Software can be used to provide alerts to the user that they have to complete a task or attend an appointment. Games, quizzes and ‘brain training’ applications can help to improve or maintain a person’s memory.

Cognitive aids are often used by the elderly to help them to live an independent lifestyle for as long as possible.

Personal emergency response systems

Personal emergency response systems, when triggered, can send an alert to a nominated mobile phone number. Different sensors can detect a range of inputs, including environmental conditions such as smoke, water and gas levels, or biometric data, including heart rate, temperature and evidence of falls. The style of sensors used can be tailored to the individual needs of the user. In addition, many such systems come with a help button installed that allows the user to call for assistance directly; they may also include cognitive aids such as alerts and reminders.

Home automation

Home automation systems employ sensors and actuators to complete day-to-day tasks in the home, such as opening and closing doors and cupboards, locking and unlocking doors and controlling energy systems. Smart home technology is also available to make your home more secure by allowing access to your CCTV stream via your smartphone, turning on all your lights if there is an intrusion, creating a ‘mockupancy’ scene that gives the impression that your home is occupied, and automatically locking your external doors at nightfall.



Denys Prykhodov / Shutterstock.com

FIGURE 5.2 Smart home technology can be controlled from your watch, tablet or phone.



Weblink
Assistive
technology

Mobility aids

Mobility aids include devices such as wheelchairs, walking frames and electric scooters. Electric wheelchairs and scooters include electric motors and use sensors to assist with navigation. They are operated with a joystick, which is controlled either by hand or with the chin, depending on the level of disability. Read more about assistive technology via the weblink.



Libby Welch/Alamy Stock Photo

FIGURE 5.3
Computerised wheelchair

Financial services

There is a wide range of technology used within the financial services industry. Traditional uses of technology in this field have included online banking, automatic teller machines and point-of-sale payment systems. As technology advances, there are new ways to use it that enhance the services offered. Buy-now, pay-later services have harnessed technology to offer customers the chance to purchase goods immediately and pay for them over the following months. These services are becoming an alternative to credit cards. At the same time, banking and financial apps allow customers to analyse their spending and saving habits to take better control of their finances.

Customer service

Financial services organisations have been early adopters of the use of **virtual assistants** for customer service purposes. In particular, **chatbots** or automated voice and text systems are increasingly being used as the initial interface with customers. As the technology used in virtual assistants improves, the efficiency of these interactions is also improving and becoming more seamless, saving time for the customer. Flowing on from this, the organisation requires fewer staff for handling customer queries, so the technology is also saving the organisation money and increasing profits.

Multi-factor authentication

A growing number of organisations are using **multi-factor authentication** when users complete financial transactions online.

Previously, a user was required to provide one form of evidence, usually the combination of a username and a password, to be able to complete a transaction online. This introduces a serious threat to security. If an unauthorised person were to gain access to that one form of evidence, they would be able to complete transactions without the knowledge of the authorised user.

Multi-factor authentication requires the user to provide two or more pieces of evidence to be able to complete a transaction online. Usually, a username and password are required along with at least one other form of evidence. Different forms of secondary evidence are explained below.

Security token

A security token is a small hardware device that is used to generate an authentication code. Often, before a user can complete a specific transaction online, they are required to generate an authentication code using the security token.



SVRSLYIMAGE/Shutterstock.com

FIGURE 5.4 Security token

Personal identification number (PIN)

A personal identification number (PIN) is used as a primary authentication technique when using an automated teller machine (ATM), but PINs can also be used as a part of a multi-factor authentication when completing transactions online. Users are often asked to set up a PIN (one that is different from the PIN they use to access an ATM) for use on online financial systems. To complete particular transactions, the PIN must be supplied along with the username and password. PINs can also be sent to the user via SMS, email or QR code verification.

SMS-based verification

Users are sometimes sent an SMS message while completing a transaction online. The SMS is sent to a pre-registered phone number. Once received, the SMS will contain details about how a transaction can be authorised. It may contain a one-off PIN or a hyperlink, or it may require the user to reply to the message.

Email verification

In a similar way to SMS-based verification, users wishing to complete a transaction online can be sent an email message containing details about how a transaction can be authorised. As the number of hoax and phishing emails is very significant, however, many financial institutions have started to avoid the use of email verification.

QR code verification

A QR code is a machine-readable code that consists of a collection of black and white squares. Each user can be given a unique QR code, which they store on their device. When

completing a financial transaction, they are required to scan the code to assist in verifying their credentials.

One-time passcode

A one-time passcode is similar to a PIN, but the user can only use the passcode for one transaction. To complete a second transaction, they will have to generate a new one-time passcode. If the user makes a mistake with the original passcode or the passcode expires, they will also need to generate a new passcode.

Secret questions

In some systems, when the user first completes their registration they are asked to set up one or more 'secret questions' and an accompanying answer to each question. When completing a transaction that requires multi-factor authorisation, they will be asked one or more of the secret questions, to which they must respond with the correct answer.

Location-based factors

The use of location-based factors involves the automated collection of data regarding the device a user is using to complete a transaction online, including the current location of the device. This data is used to identify strange patterns in the use of financial systems – in particular, if a user completes a transaction in one location and a second transaction occurs soon afterwards in a different location.

Requiring multiple pieces of evidence during the authentication stage reduces the risk of a transaction being completed by an unauthorised person.

Fraud detection

The use of technology is playing an increasing role in the detection of fraud in the financial services industry.

In the financial services industry, **fraud detection** involves identifying unauthorised transactions. Many organisations rely on data analysis and artificial intelligence to assist them in identifying transactions that may be deemed suspicious and worthy of further investigation.

Fraud detection involves viewing a transaction and comparing it to previous transactions to see if it is reasonable or consistent with previous transactions. Criteria that can be used to compare transactions can involve the amount of the transaction, the location in which it takes place, the type of purchase, the frequency of transactions or even the time of day.

'Data-mining' software is used to identify if there are any factors that are outliers to the regular transactions conducted by the user. If an outlier is identified, the organisation will investigate the transaction further, sometimes contacting the user directly for confirmation.

Artificial intelligence (AI) technology is used to 'data mine' through large data sets of previous transactions, which contain a number of transactions that are known cases of fraud or are considered suspicious. The algorithm used by the AI technology is then refined until a high proportion of the fraudulent or suspicious transactions are identified. When the developers feel confident the system is highly accurate, the technology can be used in real time to monitor transactions as they occur.

Electronic funds transfers

Electronic funds transfers (EFT) allow a payment to be made from one bank account to another using a digital system.

5.1 THINK ABOUT APPLIED COMPUTING

Which combinations of evidence do you think provide the best protection for online users of financial systems?

Data mining involves searching through existing large data sets in an attempt to find new patterns or relationships within the data.

Examples of electronic funds transfers include:

- ATM transactions
- direct deposits
- direct debits
- credit card or debit card transactions
- online bill payment
- instant payments.



Weblink
NFC technology

For credit card and debit card transactions at the point of sale, a **payment terminal** is used to facilitate the transaction. A payment terminal is a hardware device that captures credit card or debit card details and sends them to a bank for authorisation. Once authorised, funds are transferred from the customer's bank account into the organisation's bank account; sometimes these funds are transferred through an intermediate account known as a merchant account. Payment terminals can read card details using a card reader, or by using **near-field communication (NFC) technology**.

Payment applications, such as Apple Pay, Google Pay and WeChat Pay, which allow the storage of card details, can be downloaded onto smartphones, reducing the need for people to physically carry their cards with them (provided they carry their phone).

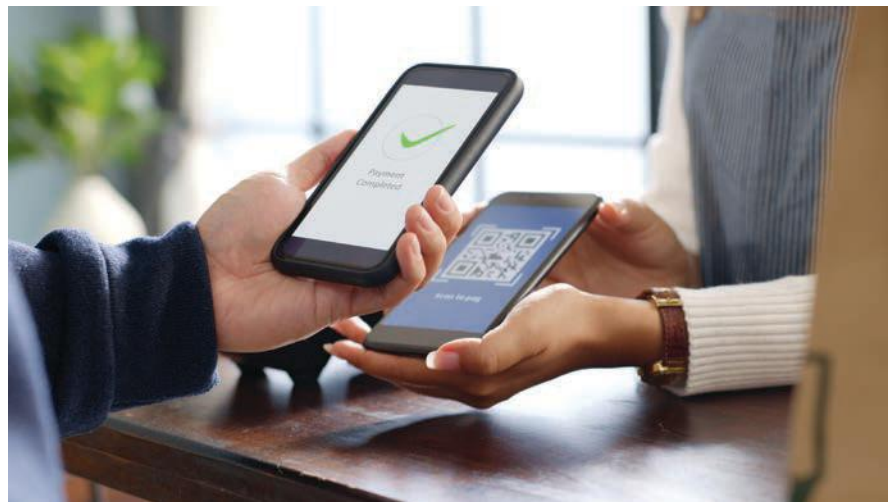
NFC technology allows contactless communication at a very close distance, without the need to configure any settings, using radio waves. Many credit cards and debit cards have a chip installed that allows the transmission of data via near-field communication technology, such as Visa payWave or MasterCard's Tap and go. There has been a massive increase in the number of contactless transactions in recent years.

Data communication between the payment terminal and the bank can be via Bluetooth, a wired network, Wi-Fi or a mobile network. Payment terminals used in aeroplanes may even use satellite communication to transfer the data. Encryption is used to protect the details of each transaction.

Many smartphones have near-field communication (NFC) technology built in. It is possible to set up NFC settings to complete a range of tasks, including sending messages to nearby devices, although to receive these messages the receiving device may need to install an NFC reader from an app store.

FIGURE 5.5

Contactless payments are now the norm.



Nattakorn_Maneerat/Shutterstock.com

Virtual currencies

Currency (or money) used to pay for goods and services is issued and controlled by a country's government and is legal tender within that country. Currency plays a key role in the economy of the country in which it was issued.

Virtual currency is unregulated, digital money, which is issued and (usually) controlled by the virtual currency's creators, and is used and accepted among the members of an online virtual community.

One type of virtual currency is **cryptocurrency**, which uses encryption to secure transactions. Bitcoin is probably the most well known cryptocurrency.

In virtual currencies, transactions occur between two users online over an internet peer-to-peer network. Each user of the currency has a virtual wallet, which stores data about the user's account. In cryptocurrencies, the wallet will contain the public and private keys used to encrypt and decrypt the transaction data. Details of each transaction are then sent to a **blockchain** to be verified and stored. A blockchain is a collection of devices connected using the internet, each device storing a copy of transactions in what is known as a block.

Users who make their devices available as part of the blockchain are rewarded in the virtual currency in which they are hosting the database. Find out more about virtual currencies and cryptocurrencies via the weblink.

Blockchain

Blockchain technology was first utilised with the introduction of Bitcoin virtual currency. A blockchain is a collection of data records (blocks), or a database, that is stored on multiple devices (chain). In some cases, the data can be stored on millions of computers.

The purpose of a blockchain is to allow data to be input and stored without allowing the data to be manipulated after storage.

When a financial transaction occurs, the transaction first needs to be verified: all the computers that store the database or block must confirm the transaction and its details. Once a transaction is verified, it is given a 'hash value'. A hash value is a value that is calculated to act as a unique identifier for that transaction in the database. Once a hash value is calculated, the data is added to the block or database on each device, along with all the previous data stored.

As the database is stored on multiple devices, it is very difficult for an individual to change the contents of any transaction because to do so they would need to access every device on the blockchain to manipulate every copy of the database. Blockchains offer security in that each new transaction is added to the top of the block, in sequential order to the previous transaction, making it very difficult for anyone to delete a specific transaction. Also, if any transaction is edited, a new record and hash value are created, setting up an audit trail that can be used to alert the users of the blockchain that a modification has occurred.

Devices that are used to store the block or database also require verification. Users who wish to store the blockchain database on their device need to complete a process known as 'proof of work'. To do this, a device needs to complete certain complex mathematical calculations to prove that it has significant processing power and functionality to be able to be part of the blockchain.

Global positioning system (GPS)

Global positioning system (GPS) uses a network of **satellites** that provide location and time data to any connected device. GPS is operated and controlled by the United States Government.

Satellites orbiting Earth on a range of different paths allow connection to the system from any location on Earth at any time. Line of sight is required, meaning natural or man-made obstacles may affect the signal.



Weblink
Virtual
currencies and
cryptocurrencies

The user is not required to transmit any data when using GPS. Rather, each device with GPS software installed acts as a receiver, which only receives data from the system. Devices that receive GPS data are known as GPS receivers.

Each GPS satellite transmits data regarding its location, time and date using radio waves. As radio waves travel at a constant speed, the time taken for each signal to be received by a receiver indicates how far away it is from the satellite. To determine the precise location of a receiver, triangulation is required. The receiver receives data from multiple satellites and uses an algorithm to calculate its location. The accuracy of the location identified by the GPS receiver may vary depending on a number of factors, but usually it is accurate to within 10 metres.

GPS does not require the services of any other network. The user does not need to be connected to the internet or to a mobile network to receive GPS data. GPS software is now installed in most smartphones, allowing individuals and organisations to harness the power of the technology.

The most obvious use for GPS technology is to allow a user to determine their location. This could apply to an individual trying to orient themselves in a new city, a ship traversing the ocean or a pilot determining their location while mid-flight.

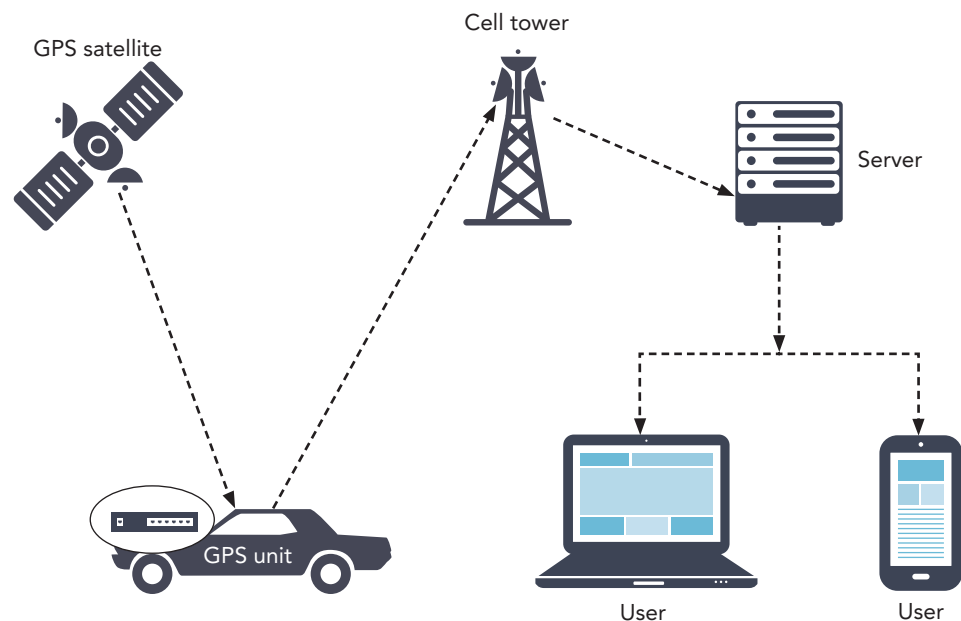


FIGURE 5.6 GPS in your car works by receiving information from satellites about location. This is sent to a phone tower and onto the internet and to your computer or phone.

Using location data, the navigation required to get from the current location to a specific destination can be determined. GPS data can be used in conjunction with mapping software to display the route a vehicle should take to get to a chosen destination. Often, multiple routes are identified and, by using a combination of live traffic conditions and historical data, the approximate time each route will take can be determined. This also applies to a range of other transport options.

GPS also makes it possible for organisations and individuals to track the progress of a person, a vehicle or an item. Many car hire organisations allow authorised users to view the tracking of a person's ride. Businesses can track the progress of a delivery van and use this information to inform customers of an estimated time when their delivery may arrive. Software applications are used to track the location of a user's smartphone in case it has been lost or stolen.

GPS plays an important role in the development of new maps. Location data in the form of latitude and longitude coordinates can be downloaded.

An often-overlooked functionality of GPS is to deliver the precise time. Each satellite has an atomic clock installed, and each time it transmits data, it includes the current time and date. Whenever a GPS receiver collects data and identifies its position, it also calculates the precise time and date of that location using the data from each satellite.

Robotics

A **robot** is a device that can complete a specific task with autonomy. Robotics incorporates intellectual property from a range of different fields, including engineering, which is related to the design and building of robots, and computing, which is related to creating the information systems used to control a robot. The range of situations in which robots are used is constantly increasing. Robots are now used in the agriculture, manufacturing, medical and military industries, to name just a few.

There are many types of robots, but they usually have the three following elements:

- a physical structure designed to achieve a particular task
- electrical components that power and control the mechanism
- programming code.

Components usually found in robots include a power source, which could range from mains electricity, batteries or compressed gas to hydraulics or solar power. Actuators are used to control and move the device. The role of an actuator is to convert the storage energy from the power source into motion. An actuator commonly found in robotic devices is an electric motor that causes a wheel to rotate, creating movement within the device.

Sensors are often used in robots to collect input about their working environment. This allows robots to perform their tasks and gives them the ability to make changes in response to any changes in their situation. Sensors may detect touch, sound, visual information or even temperature.

Some robots are used to manipulate objects, and these robots are often produced with arms and hands that allow them to perform the task for which they were designed. The hands of a robot are referred to as the end effectors, while the arm is referred to as the manipulator.

The actuators and sensors within a robot are connected to its microcontroller, which is used to control its actions. Software applications are written using a programming language; these provide the instructions needed by the robot. Some robots, such as a robot vacuum cleaner, are programmed to perform stand-alone tasks (Figure 5.7); other robots, such as a drone, can be controlled by user input via a device like a remote control.



FIGURE 5.7 A robot vacuum cleaner uses mains electricity to charge a storage battery. Touch sensors and infra-red sensors direct its route around the house.

Traffic management

Australia's population continues to increase, particularly in the capital cities. This is leading to an increase in the number of cars on the road. The procedures and processes used to efficiently and effectively control the flow of vehicles on the roads are known as **traffic management** strategies. State governments, which have responsibility for traffic management, are turning to technology to help them. Technologies such as speed cameras, red-light cameras and point-to-point and bus lane camera systems play an important role in the field of law enforcement.

Red-light cameras often use in-road sensors to detect if a car has passed through an intersection after the light has turned red. The sensors are installed on individual lanes on a road so as to be able to detect the exact lane in which a car crossed the stop line after the light had turned red. When an infringement is noticed, the sensor will trigger the camera, which is installed above ground and connected using wires, to take an image of the offending vehicle.

Speed cameras transmit a radar beam towards oncoming traffic. As the beam hits a car, signals are sent back to the camera, which then calculates the speed at which the vehicle is travelling.

In point-to-point camera systems, a number of cameras are set up at various set locations along a road. As a vehicle passes a camera, an image is taken and the time is recorded. As the vehicle passes the next camera, another image is taken and the time is again recorded. The average speed of the vehicle is then calculated by dividing the distance between the two cameras by the time taken.

Cameras are also used to read vehicle number plates. As a number plate is recorded, details are retrieved from a number of databases to check if there are any known concerns about either the driver or the vehicle. The data stored and accessed in this way could include the owner's current licence status, whether the car is insured and registered, and if the owner has any outstanding traffic offences.

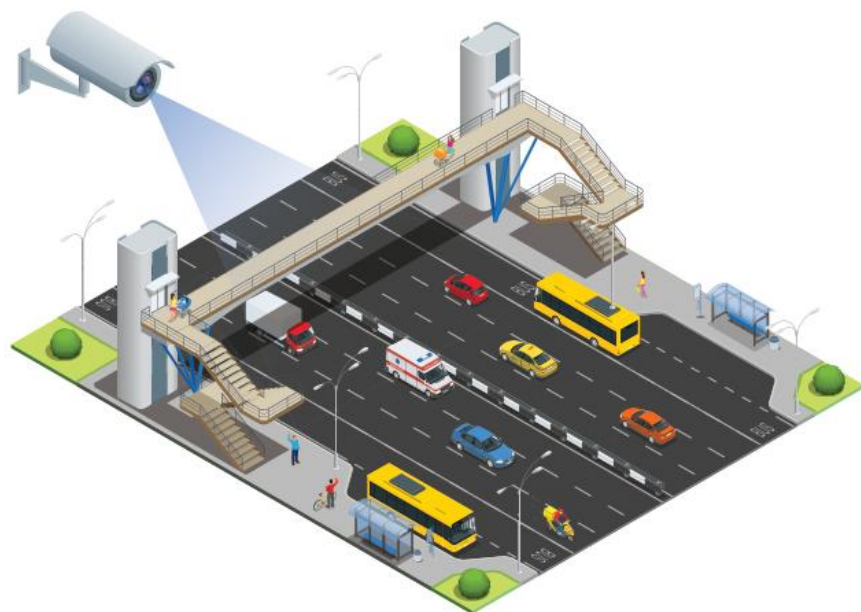


FIGURE 5.8 Cameras are used in traffic management systems.

Golden Sikorka/Shutterstock.com

Cameras can also be set up to capture images of the passenger compartment within a vehicle to check that drivers are not breaching any road rules, such as by using a mobile phone while driving.

Traffic management systems also monitor and control the flow of traffic on the roads. Sensors and cameras can detect how busy sections of roads and highways are. This data is transmitted to a central server, where decisions can be made to decrease or increase the traffic limit in some areas, or be used to alter or synchronise a collection of traffic lights that will improve the flow of traffic in that area. In some countries, once traffic incidents are detected, the system is able to broadcast traffic warnings over traditional AM/FM radio networks, or send messages and real-time traffic updates to in-car GPS systems, to alert drivers to avoid particular areas.

Emerging technologies

During your time in Applied Computing, you may find significant change occurring in the technological landscape of contemporary society. Between 2020 and 2024, for example, there have been notable transformations, such as the rapid advancement and integration of artificial intelligence, cloud computing and remote work technologies. These developments have revolutionised various industries, including education, where digital platforms and tools have become essential.

When exploring potential topics for innovative solution projects in Unit 2, Outcome 1, it would be useful to consider the following topics:

- artificial intelligence, machine learning or neural networks
- assistive and wearable technologies or **Internet of Things (IoT)**
- smart devices in health care
- drones, microcontrollers, nanosatellites and robotic devices
- games development or multimedia programming
- mixed realities, such as augmented or virtual reality
- blockchain.

This is not an exhaustive list of **emerging technologies**, just a suggestion and a starting point.

Artificial intelligence, machine learning, neural networks

Artificial intelligence (AI), **machine learning (ML)** and **neural networks** are revolutionising industries by enabling systems to learn from data, make predictions and improve over time without explicit programming.

Emerging technologies include more sophisticated AI models that can understand and generate human-like text, advanced image and speech recognition systems and autonomous vehicles.

Innovations in AI are enhancing decision-making in fields such as finance, health care and logistics, promising smarter, more efficient processes.

In the last few years, we have seen a shift towards online learning environments and the incorporation of AI-driven tools like ChatGPT to enhance teaching and learning experiences. This has resulted in arguably more flexible and personalised education but has also necessitated continuous adaptation to new technologies for teachers and students alike to stay current and effective in the classroom.

**CASE STUDY****AlphaFold**

In 2020, DeepMind, a subsidiary of Google, made a breakthrough in protein folding prediction with its AI system, AlphaFold. This AI model significantly improved the accuracy of predicting protein structures, a critical aspect of biological research that has implications for understanding diseases and developing new medications. AlphaFold's success demonstrates the potential of AI to solve complex scientific problems that were previously thought to be unsolvable. Read more about it on the Google Deep Mind website by navigating to the Technologies tab (see weblink).



Weblink
AlphaFold

Assistive and wearable technologies or Internet of Things (IoT)

Assistive and **wearable technologies**, along with IoT, are transforming how we interact with the world around us. Innovations in this space include smart watches and fitness trackers that monitor health metrics, smart home devices that automate daily tasks, and IoT-enabled medical devices that provide real-time health monitoring. These technologies are enhancing the quality of life for individuals with disabilities and chronic conditions, while IoT networks are creating more interconnected and responsive environments.

**CASE STUDY****Dexcom G7**

The Dexcom G7 is a wearable device that continuously monitors blood glucose levels for people with diabetes. It uses IoT technology to send real-time glucose data to a smartphone app, allowing users to track their levels and receive alerts for high or low readings. This innovation has greatly improved diabetes management, reducing the need for frequent finger pricks and helping users maintain better glucose control. Read more about it on the Dexcom website by navigating to the Products tab (see weblink).



Dulin / Shutterstock.com

FIGURE 5.9 A woman with type 1 diabetes uses the Dexcom G7, which is glued to one of her arms. It is waterproof, so it can be worn while swimming or bathing.

Smart devices in health care

Smart devices in health care are leading to more personalised and efficient patient care. Innovations include wearable health monitors that track vital signs, smart pill dispensers that ensure medication adherence and AI-powered diagnostic tools that assist in early disease detection. These technologies are enabling remote patient monitoring, reducing hospital readmissions and providing valuable data for medical research.

Philips HealthSuite

Philips HealthSuite is a cloud-based platform that integrates data from various smart health devices, such as wearable monitors, connected medical devices and patient health records. It enables healthcare providers to monitor patients remotely, analyse health data and make informed decisions. This platform has been used to improve patient outcomes and streamline healthcare processes, especially in managing chronic conditions like heart disease and diabetes.

Read more about it on the Philips website by navigating to the Professional healthcare tab, then clicking on Solutions, followed by the HealthSuite Digital Platform link (see weblink).

CASE STUDY



Weblink
Philips
HealthSuite

Drones, microcontrollers, nanosatellites and robotic devices

Drones, microcontrollers, nanosatellites and robotic devices are driving advancements in automation, data collection and exploration. Emerging technologies include autonomous drones for delivery and surveillance, microcontrollers that enable smart devices and robotics, nanosatellites for low-cost space research and robotic devices that perform complex tasks in manufacturing and health care. These innovations are expanding capabilities in areas such as logistics, environmental monitoring and space exploration.



Hydronalix

FIGURE 5.10 This mobile robot rescues distressed swimmers by scanning underwater movements with sonar, and enabling lifeguards on shore to provide instructions and reassurance to the person being rescued.

**CASE STUDY****Weblink**
Zipline

Zipline

Zipline operates a drone delivery service that transports medical supplies to remote and hard-to-reach areas. In Rwanda and Ghana, Zipline's drones deliver blood, vaccines and medications to hospitals and clinics, significantly reducing delivery times from hours to minutes. This service has been crucial in improving access to essential medical supplies and has saved countless lives by ensuring timely deliveries.

Read more about it via the Zipline weblink.



Joerg Boethling / Alamy Stock Photo

FIGURE 5.11 The Zipline safely delivers medications and supplies.

Games development or multimedia programming

Games development and multimedia programming are being revolutionised by advancements in graphics, AI, and interactive storytelling. Innovations include more immersive and realistic virtual worlds, AI-driven characters that adapt to player behaviour, and cross-platform game engines that streamline development. Emerging technologies such as ray tracing for lifelike lighting and cloud gaming services are enhancing the gaming experience, making high-quality games more accessible.

**CASE STUDY**

Unity

Unity is a widely used game development platform that has significantly impacted the indie game development scene. Its accessible interface and robust features have allowed small teams and individual developers to create high-quality games without needing extensive resources. Unity's cross-platform capabilities enable developers to publish their games on multiple platforms, including PC, consoles and mobile devices, thereby reaching a broader audience and fostering innovation in the gaming industry.



Niantic, Inc.

FIGURE 5.12
Augmented reality results when a computer-generated image overlays, or augments, the real world.

Mixed realities

Mixed realities, including **augmented reality (AR)** and **virtual reality (VR)**, are transforming how we interact with digital content. Innovations in AR and VR are creating more immersive and interactive experiences in gaming, education, training and remote collaboration. Advances include lighter, more comfortable headsets, improved motion tracking and more realistic graphics. These technologies are enabling new forms of communication, learning, and entertainment by blending the physical and digital worlds.

Lightship Augmented Reality Developer Kit (ARDK)

In 2022, Niantic, the company behind *Pokémon GO*, launched the Lightship Augmented Reality Developer Kit (ARDK). This platform provides developers with tools to create advanced AR experiences that blend the digital and physical worlds seamlessly. Lightship ARDK includes features like real-time mapping, multiplayer functionality and semantic segmentation, which allows apps to understand the physical environment in more detail. This technology has been utilised by various developers to create interactive and immersive experiences in gaming, education and tourism. For example, the game *Peridot*, an AR pet simulation game, leverages Lightship's capabilities to create lifelike interactions and environments for virtual pets.

**CASE
STUDY**



Blockchain was discussed earlier in this chapter, on page 185.

Blockchain

Blockchain technology is redefining how data is stored, shared and secured. Innovations in blockchain include decentralised finance (DeFi) platforms, non-fungible tokens (NFTs) for digital ownership and smart contracts that automate transactions. Emerging technologies are enhancing the scalability and efficiency of blockchain networks, enabling secure, transparent and tamper-proof record-keeping in industries such as finance, supply chain management and health care.



CASE STUDY

Microsoft Entra

In 2023, the ID2020 Alliance, in collaboration with Microsoft, expanded its decentralised identity solution built on the Entra (formerly Azure) blockchain. This initiative leverages blockchain technology to create secure, self-sovereign digital identities. The system enables individuals to own and control their personal information without relying on centralised databases. Users can store their identity data on a blockchain, granting access only when necessary and under their terms.

One practical application has been in providing legal identities to refugees and displaced persons. These digital identities ensure that individuals can access essential services such as health care, education and financial services, regardless of their geographic location. By securing identity data on the blockchain, ID2020 ensures that the information is unchangeable and tamper-proof, significantly reducing the risk of identity theft and fraud. This project represents a significant step towards empowering individuals with their digital identities, promoting privacy and enhancing access to essential services.

Read more about Microsoft Entra and about how blockchain can change lives via the weblinks.



Weblinks
Microsoft Entra
Blockchain
article

Impact of emerging technologies

Emerging technologies are capable of making significant change within society. These changes can be positive and negative. In Unit 2, Outcome 1, you are required to create an innovative solution to a need or opportunity. As part of this outcome you are required to discuss potential issues that may arise due to the development of the innovative solution.

Issues and impacts can be related to:

- automation
- cyberbullying
- productivity
- economic issues
- the decline of human interactions and interpersonal skills.

Automation

The term **automation** refers to processes that can be completed without the need for human intervention. Automation can be used to operate machinery; capture, communicate and store data and information; control and steer cars, ships, aircraft and other transport vehicles; or in the manufacture of goods and products.



Bloomberg/Getty Images

FIGURE 5.13

Automation is widely used in the mining industry.

Control of an automated system can be as simple as an on–off switch, or it can be as complicated as a complex algorithm containing a range of control and processing structures and multiple variables, and using a range of complex hardware devices.

Impact of automation

An automated system may be complex and expensive to install. It is often expected that, over the long term, once a process is automated, the system will be more efficient and effective and will produce consistent results and save the organisation money. In the short term, however, setting up the system can be costly and it may take time for the system to be fine-tuned to become fully effective.

Automation has an impact in deskilling the labour force (see pages 196–7) and creating job losses (see pages 197–8). A range of new jobs may be created that require workers with new skills. Existing workers may have to be retrained. One question to ask is how the education system will cope with training students for a range of jobs that may not yet exist.

Automation needs a digital system to be designed and programmed to complete specific tasks. If the task changes – to a new task or to one it was not originally designed to do – it can be difficult, expensive and time-consuming to reprogram the system. In this situation, human workers may still need to be used to complete these activities.

In the manufacturing industry, part of the reason for using automation is to be able to produce products of a consistent quality. This has, however, reduced the individuality of manufactured products. Consistency may be achieved, but the quality may not be the same as that achieved by a human craftsperson.

Cyberbullying

Cyberbullying occurs when a digital system is used to threaten, harass or intimidate another individual. This could be in the form of lying, spreading rumours, playing cruel jokes or making threats. Its intent is to cause harm to another person or a group of people.

Bullying existed in various forms before the digital age. However, the anonymity and accessibility of the internet has been a major contributing factor in the increase in bullying.

5.2 THINK ABOUT APPLIED COMPUTING

List the impacts of automation of the workforce, and add two more that have not been discussed here.



Although in many cases a cyberbully may know their target, it is not always the case that the individual being bullied knows who is responsible.

The increased use of smartphones has led to an increase in cyberbullying as online platforms and social media have become more accessible to users. Features of a smartphone such as social media apps, internet access and a camera have contributed to an increase in bullying using emerging technology.

People posting messages using electronic communication often feel more detached, and less responsible for their actions, than if they were communicating the same message face to face. Once messages are posted online, the potential audience who can view them is unlimited and they are very difficult to delete.

Cyberbullying can cause stress and anxiety for the targeted individual. This can result in an individual becoming less productive, which also affects their social life, work life and the organisation in which they function.

Economic issues

Economics is the study of the production, distribution and consumption of goods and services. It involves the study of scarcity and choice, based on the idea that the demand for resources (e.g. money, supplies, equipment) is greater than the resources that are available. Decisions need to be made about how these resources are to be allocated and used. These decisions sometimes lead to negative consequences for some stakeholders. These are referred to as economic issues.

Economic issues involving emerging technologies can include:

- deskillling
- job loss
- misuse.

Deskilling

Deskilling occurs when a skilled worker is replaced, or their job specification is reduced, as a result of the implementation of technology. In any given industry, this can result in fewer workers having the skills to complete specific tasks, and can lead to existing jobs being eliminated altogether. In society more broadly, this can result in an overall reduction in knowledge and can have negative effects on culture and lifestyles.

In manufacturing, for example, a significantly reduced number of tasks may now require workers to complete them as robots are installed to complete other tasks. Trained tradespeople and craftspeople may no longer be required to apply the skills that they were trained in, and may focus instead on the limited number of tasks that robotic devices are unable to complete.

Technology is introduced in order to achieve an increase in productivity, maintain product consistency and reduce the overall cost of the production process. The downstream effect of this could be that workers become less engaged and enthusiastic about their roles, and the quality of the products manufactured becomes more uniform and less personalised.

There are a range of traditional jobs and roles in which job specifications have been altered or reduced following the introduction of automation. Roles have been replaced or redefined in the field of art and design as the use of software applications in design work becomes commonplace. Similarly, 3D printing can be used to produce a wide range of products that were previously designed and manufactured by highly skilled workers. This may eventually lead to age-old skills involved in manually designing and creating products being lost forever.



Alexander Tolstykh/Alamy Stock Photo

FIGURE 5.14

Automated pilot systems are capable of completing many of the tasks once completed by human pilots.

In the commercial aviation industry, the number of tasks that are conducted using a digital system has increased significantly, and the number completed by a human pilot has been correspondingly reduced. Automated pilot systems can now conduct up to 90 per cent of the tasks that are required in the time between when an aeroplane leaves the gate at one airport and when it arrives at the gate at the destination airport.

The use of AI, GPS, virtual assistants and IoT devices allows conditions to be monitored, decisions to be made and tasks to be completed without the need for human intervention. Human pilots are still required to monitor the automated system and complete tasks that cannot be automated, and are able to override decisions when they deem it appropriate. It is not uncommon for pilots of large commercial aircraft to spend some time each year flying small, less sophisticated aircraft that contain less automation. This ensures that they do not lose the skills of flying, such as taking off and landing, that are often performed by autopilots on regular commercial flights.

Job losses

An economic issue that overlaps with deskilling is that of job losses. A **job loss** occurs when a job formerly completed by a human in an organisation is no longer required. This results in the worker becoming redundant (excess to needs). Sometimes, a job loss occurs as a direct result of new technology being introduced in the organisation.

Examples of the use of technology leading to job losses include the use of automatic teller machines, electronic funds transfer and online banking. These have reduced the number of financial transactions that require the involvement of a human worker, and have thus reduced the demand for bank staff. One of the reasons for the increasing popularity of cryptocurrencies is that they avoid traditional banking systems; the growth in cryptocurrencies may also affect the demand for traditional banking products and banking staff.

Automatic checkout systems at supermarkets and department stores are reducing the number of situations in which a skilled human worker is needed. The growing use of virtual assistants and chatbots to interact with customers is replacing the need for customer service and call centre staff.

5.3 THINK ABOUT APPLIED COMPUTING

List 10 skills that you think might be lost due to the introduction of technology.

**THINK ABOUT
APPLIED COMPUTING****5.4**

Research the types of tasks that businesses often send offshore. What technology do businesses use to allow offshore workers to complete these tasks?

The use of robots in the manufacturing process is well documented. The widespread use of smart devices in businesses has reduced the number of tasks that require a human to complete them. Smart refrigerators have the ability to keep track of items added to or removed from them, and can use this information to create shopping lists and place online orders. The use of smart refrigerators in cafes and restaurants may not lead directly to job losses, but it has the effect of reducing the number of tasks staff must complete. This, combined with other efficiencies related to the use of technologies (e.g. the use of automated coffee machines), may lead to a reduction in the number of staff needed.

Drones are used in the agricultural industry for a range of applications, including crop spraying, monitoring of crops and irrigation systems and aerial photography. This is reducing the need for skilled agricultural workers. The use of assistive technology is allowing elderly and disabled people to live a more independent lifestyle, thus reducing the demand for skilled medical and nursing staff.

As technology and communication methods improve, more organisations are completing jobs offshore. This allows them to employ workers in other countries, where wages may be lower, to complete tasks that once were completed by workers in their home country. A challenge for these organisations and countries is to reskill and retrain their existing workforce in areas of new employment; otherwise, these workers may become excess to needs.

Misuse

Emerging technologies, while offering numerous benefits and advancements, also pose risks of misuse. Misuse can arise intentionally or inadvertently and the consequences can be wide-ranging, affecting individuals, societies and even global systems.

One concern about emerging technologies is the potential for misuse that results in economic implications. For example, the deployment of automation and AI technologies in a way that prioritises short-term productivity gains can result in long-term economic instability and reduced workforce development.

Imagine a scenario where a majority of industries rapidly adopt advanced automation technologies to cut costs and increase efficiency. In the short term, this leads to significant productivity boosts and cost reductions for businesses. However, this widespread adoption of automation is done without adequate investment in workforce retraining or the creation of new employment opportunities for those displaced by machines.

As a result, large segments of the population find themselves unemployed or underemployed, lacking the skills needed for the few jobs that are available. This leads to increased economic inequality, as the benefits of technological advancements are concentrated among highly skilled workers, while low- and middle-skilled workers face job scarcity. Consumer spending decreases due to widespread job loss and economic insecurity, which in turn affects demand for goods and services.

Furthermore, the rapid and unregulated deployment of such technologies could lead to market monopolies, where a few large corporations control vast amounts of the economy, stifling innovation and competition. Small businesses struggle to compete with these tech giants, leading to less diversity in the marketplace and fewer choices for consumers.

The misuse of technology, particularly in the form of unchecked automation without corresponding social and economic policies to support workforce transition, can lead to economic disparities, reduced consumer spending, market monopolisation, and an overall destabilisation of the economy. It underscores the need for balanced and thoughtful integration of technology into the economy, with considerations for workforce development, social safety nets and equitable access to the benefits of technological advancements.

Human interactions

By nature, humans are social creatures, who require interaction with others, whether in social situations, forming relationships, creating networks or discussing emotions and feelings. **Human interactions** involve people communicating with other people. This can be through a face-to-face conversation, a text message or a smile. Research has shown that socialising and interacting with other people is a fundamental human need along with the need for food, shelter and water.

As technology has advanced, the increased use of automation and the ability to complete tasks online using an internet connection or to work from remote locations and while travelling have led to a significant decrease in human interactions, particularly face-to-face interactions. In many ways these advances in technology have helped people stay connected with others, whether through email, social networks, video conferencing or forums and blogs. The rise in popularity of smartphones has accelerated this connectivity but, as a result, actual face-to-face human interactions are no longer necessarily as frequent as in previous years.

Impact of loss of human interaction

Evidence suggests that regular human interactions offer health benefits for individuals. A lack of human interaction has been known to lead to a range of health issues, including loneliness, depression and associated mental health issues.

Human interactions are important within organisations. They help to achieve effective collaboration between members of the organisation that cannot always be achieved through online communication. By meeting face to face, people are able to have more in-depth conversations, build relationships, and pick up non-verbal signals and body language more easily.

In the classroom, human interaction plays a major role in learning. The teacher's role in the classroom is that of a facilitator, interacting with students. Students also interact with one another through discussion and group work. With the increasing number of courses being delivered online, many of the advantages of the traditional classroom may be lost to students.

Interpersonal skills

A decrease in human interactions can lead to a reduction in interpersonal skills. **Interpersonal skills**, or so-called soft skills, are skills that are required to interact and communicate effectively with others. These skills are important in building relationships with family, friends and work colleagues. More and more organisations are team-based, so employees are expected to be able to build positive relationships with other people, such as being supportive, respectful and collaborative.

Interpersonal skills include the ability to:

- make a decision
- listen
- negotiate
- communicate verbally and non-verbally
- problem-solve.

Communication can be verbal (spoken and listened to, or written) or non-verbal, such as through eye contact, smiling, frowning, turning towards the speaker or nodding. Good interpersonal skills result from both types of communication. It is usually the non-verbal forms of communication that suffer when technology is used to communicate.

5.5 THINK ABOUT APPLIED COMPUTING

List the impacts of the loss of human interaction due to technology, and add two more that have not been discussed here.

5.6 THINK ABOUT APPLIED COMPUTING

Observe the individuals within a group – it might be in a coffee shop, a library, at home, at school or another place. Write down three different types of non-verbal communication that you see being used within that group.

5

CHAPTER SUMMARY

Essential terms

accessibility aids technological solutions designed to make everyday tasks accessible to everyone, especially to those with an impairment or disability

artificial intelligence (AI) the simulation of human intelligence processes by machines, especially computer systems

assistive technology any technology that allows an individual or organisation to complete a task that they would not normally be able to complete, or makes it easier or safer to complete day-to-day activities

augmented reality (AR) overlaying computer-generated images onto real-life images to provide a composite view

automation when processes or procedures can be completed without the need for human intervention

blockchain a collection of data records (blocks), or a database, that is stored on multiple devices (chain)

chatbot software that uses voice or text prompts to mimic human conversation

cognitive aids documents or prompts designed to assist people with a physical, cognitive or emotional impairment that affects processes such as attention, memory, navigation, planning or sequencing of events or tasks

cryptocurrency a type of virtual currency that uses encryption to secure transactions

cyberbullying the use of a digital system to bully another individual; this could be in the form of intimidation, harassment, rumours or threats

deskilling the replacement of a skilled worker, or a reduction in their job specification, occasioned by the implementation of technology

drone an aircraft without a human pilot

economics the study of the production, distribution and consumption of goods and services

electronic funds transfer (EFT) a system of transactions that allows financial payments to be made from one bank account to another, using a digital system

emerging technologies new technologies that are perceived to be capable of making significant change within society

fraud detection identification of unauthorised financial transactions

global positioning system (GPS) a network of satellites that provide location and time data to any connected device

human interactions instances of people communicating with other people

Internet of Things (IoT) collectively, the billions of devices around the world that are connected to the internet, collecting, receiving and sending data

interpersonal skills skills required to interact and communicate with others

job loss occurs when a job, formerly completed by a human, is no longer needed in an organisation

machine learning (ML) a subset of AI where computers use algorithms to learn from data and improve their performance over time without explicit programming

microcontroller an integrated circuit that incorporates a processor, memory and input/output (I/O) controllers within one chip

mixed reality a blend of physical and digital worlds, allowing real and virtual elements to interact in real time

multi-factor authentication a process that requires a user to provide two or more pieces of evidence of identity, such as a username and password combination and an SMS or email code, in order to be able to complete a transaction online

nanosatellite a small satellite that weighs between 1 and 10 kilograms

near-field communication (NFC) technology digital technology using radio waves that allows contactless communication between devices over a very small distance, without the need to configure settings; examples include payWave and myki

neural network a computational model inspired by the human brain's structure, consisting of interconnected nodes (neurons) that process data in layers to perform tasks such as pattern recognition and decision-making

payment terminal a hardware device that captures credit card or debit card details and sends these details to a bank for authorisation

robot a device that can complete a specific task with autonomy

satellite an object in space that orbits around a bigger object

smart device an electronic device that can connect, share and interact with other devices and users, often via the internet

sustainability the process of managing change in a balanced, controlled way so that other factors are not negatively affected

traffic management procedures and processes used to efficiently and effectively control the flow of vehicles on roads

virtual assistant a software agent that can perform tasks or services for an individual

virtual currency unregulated, digital money, which is issued and (usually) controlled by its creator and is used and accepted among the members of an online virtual community

virtual reality (VR) the use of computer-generated three-dimensional images and specialised equipment to allow a user to interact with a simulated environment different from the real world

wearable technology small smart devices that can be worn on, or implanted into, the body or woven into clothing

Important facts

- 1** Curiosity and ingenuity are critical elements in the inspiration of innovation, particularly when considering global challenges and aspirations for a sustainable future.
- 2** The United Nations' Sustainable Development Goals (SDGs) are a framework of 17 interlinked global goals designed to achieve a better and more sustainable future for all by 2030.
- 3** Existing technologies that are designed to make our lives easier include assistive technologies, financial services, global positioning system (GPS), robotics and traffic management.
- 4** Assistive technologies allow individuals or organisations to complete tasks that they would not normally be able to complete. They may also make it easier or safer to complete day-to-day activities.
- 5** Accessibility aids are designed to make everyday tasks accessible to everyone, especially those who have additional needs.
- 6** Cognitive aids assist people with a physical, cognitive or emotional impairment that affects processes such as attention, memory, navigation, planning or the sequencing of events or tasks.
- 7** Financial services organisations have been early adopters of technologies, such as with the use of virtual assistants, multi-factor authentication and fraud-detection technologies.
- 8** Virtual currencies are unregulated digital currencies. Cryptocurrencies use encryption to secure transactions.
- 9** Secure transactions use blockchain technology for verification and storage. A blockchain is a collection of devices connected using the internet, with each device storing a copy of transactions in 'blocks'.
- 10** Global positioning system (GPS) uses networks of satellites that provide locations and times of day to any connected device. GPS is operated and controlled by the United States Government.
- 11** A robot is a device that can complete a specific task with autonomy.
- 12** Traffic management is a group of procedures and processes that are used to efficiently and effectively control the flow of vehicles on the roads.
- 13** Emerging technologies include artificial intelligence, machine learning, neural networks, assistive and wearable technologies, Internet of Things, smart devices in health care, drones, microcontrollers, nanosatellites, robotic devices, games development, multimedia programming, mixed realities and blockchain.
- 14** Economic issues involving emerging technologies can include:
 - access
 - deskilling
 - job loss
 - misuse
 - sustainability.
- 15** Issues and impacts related to emerging technologies include:
 - automation
 - cyberbullying
 - human interaction
 - interpersonal skills.



Innovation

- 1 How do software applications contribute to the advancement of specific Sustainable Development Goals?
- 2 Why is it important for innovations inspired by the SDGs to be both technologically advanced and socially and environmentally responsible?

Emerging trends in digital systems

- 3 Explain what is meant by the Internet of Things (IoT). Describe one benefit of the IoT for organisations.
- 4 Give two reasons why a person might wear a wearable technology.
- 5 Explain how robotic devices ensure quality control.
- 6 Explain how drones are used to improve efficiency in the agricultural industry.
- 7 What is meant by the term 'a swarm of satellites'? How appropriate is this term?

Functions and capabilities of digital systems

- 8 Give two examples of assistive technology and how they are used to improve a person's quality of life.
- 9 List three uses of technology in the financial services industry.
- 10 Describe how GPS identifies a car's location.
- 11 Describe the components that are used in a robotic device.
- 12 Discuss one use of technology in traffic management.

Economic issues involving emerging technologies

- 13 List two factors that will affect a user's ability to access emerging technology.
- 14 Explain how deskilling could be a result of emerging technology.
- 15 Discuss the relationship between automation and possible job loss.
- 16 Explain how mobile phone production could be made more sustainable.

Impact of current and emerging technologies

- 17 Describe two ways that automation could affect a worker in the car industry.
- 18 What is cyberbullying? How does technology contribute to it?
- 19 Explain why human interactions are important.
- 20 Give three reasons why good interpersonal skills are important in a workplace.



APPLY YOUR KNOWLEDGE

Answer questions 1–3 below for each of the following emerging technologies:

- artificial intelligence, machine learning or neural networks
- assistive and wearable technologies or Internet of Things (IoT)
- smart devices in health care
- drones, microcontrollers, nanosatellites and robotic devices
- games development or multimedia programming
- mixed realities, such as augmented or virtual reality
- blockchain.

1 Explain how the technology works.

2 Discuss trends in the use of the technology by individuals or organisations.

3 Discuss economic, legal and social issues related to the use of that technology.

6

Development and evaluation of an innovative solution

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

- KK2.1.4: characteristics of creative and innovative solutions
- KK2.1.5: solution specifications
- KK2.1.6: methods for collecting data to determine user needs and requirements
- KK2.1.7: design tools and techniques for representing solution designs
- KK2.1.8: techniques for developing innovative solutions
- KK2.1.9: techniques for documenting the development of solutions, task delegation and monitoring project progress
- KK2.1.10: techniques for testing and evaluating innovative solutions
- KK2.1.11: how emerging technologies are affected by key legislation and frameworks
- KK2.1.12: ethical issues arising from the development of emerging technologies
- KK2.1.13: ethical issues arising from the use of artificial intelligence
- KK2.1.14: evaluation criteria and techniques for evaluating the efficiency and effectiveness of innovative solutions.

Victorian Curriculum and Assessment Authority, Victorian Certificate of Education Applied Computing Study Design 2025–2028, VCAA, 2024, pp. 32–3

FOR THE STUDENT

This chapter, along with Chapter 5, contains information that is required for Unit 2, Area of Study 1. For this outcome, you are required to work as a member of a team to develop an innovative solution to a need or opportunity related to the use of an emerging technology, following the problem-solving methodology. In this chapter, you will find information about the laws and other issues related to the use of emerging technologies. The steps involved in the problem-solving methodology are discussed, starting with data collection and continuing through to evaluation. These steps need to be demonstrated as part of the outcome for Unit 2, Area of Study 1.

This chapter discusses issues related to the use of emerging technologies, including legal, economic and ethical issues. Then there is discussion of the stages of the problem-solving methodology, which you need to demonstrate as part of the outcome. At the completion of Chapters 5 and 6, you should be able to demonstrate the key skills required for Unit 2, Outcome 1, where, working in groups, you create an innovative solution to a need or opportunity using an emerging technology.



Key legislation and emerging technologies

Legislation is the body of laws, created by governments, that individuals and organisations in each government's jurisdiction must follow. The introduction of new technologies presents significant challenges for governments as well as for developers and users of the emerging technology.

By their nature, emerging technologies have the potential to cause significant change in society. Each emerging technology – for example, artificial intelligence, recreational drones, nanosatellites and virtual assistants – can provide new and potentially unseen functionality and features. Governments must be able to respond quickly to create, modify or enforce existing legislation to ensure that these technologies uphold the values of the society in which they function. Equally, the developers and users of emerging technology need to ensure that they are not breaching any existing laws in their government jurisdiction.

In Australia, there are three levels of government. The **federal parliament** is responsible for creating legislation that applies to all individuals and organisations within Australia. **State parliaments** are responsible for creating legislation that applies to all individuals and organisations within a particular state, such as New South Wales or Victoria, while **local councils** create by-laws that apply within their individual local council area.

Both federal and state parliaments create new laws by introducing an Act of Parliament. An **Act of Parliament** is a document that contains a set of conditions and rules about a particular area or topic. Once parliament passes an Act, it then becomes law in that jurisdiction.

Acts of Parliament (or pieces of legislation) that have an effect on emerging technologies include:

- *Privacy Act 1988* (Commonwealth)
- *Privacy and Data Protection Act 2014* (Victoria)
- *Health Records Act 2001* (Victoria)
- *Copyright Act 1968* (Commonwealth)
- Australia's Artificial Intelligence (AI) Ethics Principles.

Privacy Act 1988

The **Privacy Act 1988** is a federal law. It contains conditions and rules related to how and when personal data and information can be collected by Australian government agencies and private organisations that turn over more than \$3 million annually. The Privacy Act was designed to be technologically neutral, but a great deal has changed since 1988 and the vast advances in emerging technologies will seriously challenge the current Act in the future.

Australian Privacy Principles

There are 13 Australian Privacy Principles (APPs), described in Table 6.1. For the purposes of Applied Computing, the focus is on APPs 1, 6 and 11. These are marked with an asterisk.

TABLE 6.1 Australian Privacy Principles

Principle	Description
*APP 1 – Open and transparent management of personal information	Organisations must have a clearly articulated and accessible privacy policy. This policy should be freely available and easy for individuals to find. It should explain how and why the organisation collects, holds, uses and discloses personal information. The policy must also outline how an individual can access and correct their personal information held by the organisation.
APP 2 – Anonymity and pseudonymity	If an individual chooses not to provide their personal information, the organisation should not enforce identification unless it is necessary for the specific service. Where practicable and lawful, individuals have the option to use a pseudonym instead of their actual name. A pseudonym is a name or other identifier that an individual uses instead of their actual name.
APP 3 – Collection of solicited personal information	Organisations can only collect personal information that is reasonably necessary for their functions or activities. They must also notify individuals about the purpose of collection and other relevant details. Individuals should be aware of why their information is being collected and how it will be used.
APP 4 – Dealing with unsolicited personal information	Organisations must destroy or de-identify the information as soon as practicable if the information was not requested or asked for. There should be no use or disclosure of unsolicited personal information unless it is permitted by law.
APP 5 – Notification of the collection of personal information	Requires organisations to inform individuals at or before the time of collecting their personal information about key details, including that their information is being requested, why it is being requested, potential recipients of the information, legal requirements or consequences if the information is not provided.
*APP 6 – Use or disclosure of personal information	Organisations are allowed to use or disclose personal information for the primary purpose it was collected. They may also use or disclose personal information for secondary purposes if the individual has consented or if it is otherwise permitted by law.
APP 7 – Direct marketing	Organisations must not use or disclose personal information for direct marketing purposes without the individual's consent. They should provide a clear and easy way for individuals to opt out of direct marketing communications.
APP 8 – Cross-border disclosure of personal information	Before disclosing personal information to an overseas recipient, organisations must take reasonable steps to ensure that the recipient will protect the information in a manner consistent with the Australian Privacy Principles. Individuals should be informed about the likelihood of their information being sent overseas and the countries or regions where recipients are located.
APP 9 – Adoption, use or disclosure of government related identifiers	Organisations are generally prohibited from adopting, using or disclosing government related identifiers as their own. A government related identifier is a unique identifier assigned by a government agency, such as a tax file number or driver's licence number.
APP 10 – Quality of personal information	Organisations are required to take reasonable steps to ensure that the personal information they collect is accurate, up to date and complete. This includes verifying the information at the time of collection and regularly reviewing and updating it if necessary. Individuals have the right to request correction of their personal information if they believe it is inaccurate, incomplete or out of date, and organisations must respond to such requests promptly.
*APP 11 – Security of personal information	Organisations are required to take reasonable steps to protect the personal information they hold from misuse, interference, loss, unauthorised access, modification or disclosure. This includes implementing security measures such as encryption, access controls and secure storage to prevent unauthorised access and data breaches. Organisations must also have processes in place to detect and respond to data breaches, including notifying affected individuals and relevant authorities when required.
APP 12 – Access to personal information	Individuals have the right to request access to their personal information held by an organisation. Organisations must provide access to the requested information within a reasonable time and at no cost.
APP 13 – Correction of personal information	Individuals have the right to request corrections to their personal information if they believe it is inaccurate, out of date, incomplete or misleading. Organisations must take reasonable steps to correct the information upon receiving such a request, unless there are valid reasons not to do so. If an organisation refuses to correct the information, they must provide reasons for the refusal.

CASE
STUDY

Emerging technologies and privacy

Technology is not the enemy of privacy. Technology can be privacy enhancing.

...
Developments in biometric technologies have been at the forefront of this change. Back when the Privacy Act was introduced in 1988, many biometric technologies were largely confined to science fiction movies. Of course, a few, such as the use of fingerprints in law enforcement, were well established. However, the concept that biometric technologies could become part of our everyday consumer transactions was almost unthinkable.

A person standing in line at a bank branch in 1988 would struggle to conceive a future where they could phone their bank, be identified by voice recognition technology, and transact from the comfort of their own home. Yet today, this is a reality.

A worker signing a time sheet as they arrived at work in 1988, would struggle to conceive a time when they would be required to have a fingerprint scanned to clock on. Yet for some people today, this is a reality.

A young adult entering a nightclub in 1988 would struggle to conceive a future where they would have to submit to a face scan before being allowed entry. This would have been the crazy plot of some futuristic television show. But today, this is also a reality.

We are likely to continue to see an increasing use of biometric technologies like those I have just mentioned, as well as iris scanning, palm scanning, and many others, in ways that we cannot predict. Assuming that these new technologies are developed in a way that is genuinely sensitive to privacy, this need not necessarily be a bad thing.

What is interesting about biometric technology is that we tend to hear both that it is good and bad for people's privacy.

... For example, voice recognition technology is being rolled out in some call centres to identify callers, leading to more effective protection of clients' personal information.

On the other hand, we hear that biometric technology has the potential to invade our privacy. For example, in the film *Minority Report*, individuals confront ubiquitous iris scanning infrastructure and technology which allows their every activity to be tracked.

How do such obviously divergent views on privacy and biometrics coexist?

The answer is: because biometric technology is what we make it. Biometric technologies are not inherently good or bad for privacy, and privacy is not a blocker to the use of biometric technologies. These technologies can become good or bad for privacy depending on how they are designed, developed and deployed.

Pilgrim, T. (27 May 2010). Privacy in Australia: Challenges and opportunities [Speech], Office of the Australian Information Commissioner website – www.oaic.gov.au CC BY 3.0 Au

THINK ABOUT APPLIED COMPUTING

6.1

Think of one other emerging technology. How does the Privacy Act 1988 affect it?

Privacy and Data Protection Act 2014

The **Privacy and Data Protection Act 2014** is a Victorian state law. It contains conditions and rules related to how and when personal data and information can be collected by Victorian government agencies, including local councils, and private organisations that conduct contract work on behalf of the Victorian Government and councils.

The *Privacy and Data Protection Act 2014* is based on 10 Information Privacy Principles (IPPs), which, although different, do have some similarities to the Australian Privacy Principles (APPs).

An abridged version of the Information Privacy Principles is shown in Table 6.2. For the purposes of Applied Computing, the focus is on IPPs 1, 2, 5 and 8. These are marked with an asterisk.

TABLE 6.2 Information Privacy Principles

***IPP 1 Collection**

An organisation can only collect personal information if it is necessary to fulfil one or more of its functions. It must collect information only by lawful and fair means, and not in an unreasonably intrusive way. It must provide notice of the collection, outlining matters such as the purpose of collection and how individuals can access the information.

***IPP 2 Use and disclosure**

Personal information can only be used and disclosed for the primary purpose for which it was collected, or for a secondary purpose that would be reasonably expected.

IPP 3 Data quality

Organisations must keep personal information accurate, complete and up to date. The accuracy of personal information should be verified at the time of collection, and periodically checked as long as it is used and disclosed by the organisation.

IPP 4 Data security

Organisations need to protect the personal information they hold from misuse, loss, unauthorised access, modification or disclosure. An organisation must take reasonable steps to destroy or permanently de-identify personal information when it is no longer needed.

***IPP 5 Openness**

Organisations must have clearly expressed policies on the way they manage personal information.

IPP 6 Access and correction

Individuals have the right to seek access to their own personal information and to make corrections to it if necessary. An organisation may only refuse in limited circumstances that are detailed in the PDP Act, for example where disclosure might threaten the safety of an individual.

IPP 7 Unique identifiers

A unique identifier is an identifier (usually a number) that is used for the purpose of identifying an individual. Use of unique identifiers is only allowed where an organisation can demonstrate that the assignment is necessary to carry out its functions efficiently.

***IPP 8 Anonymity**

Where lawful and practicable, individuals should have the option of transacting with an organisation without identifying themselves.

IPP 9 Transborder data flows

If an individual's personal information travels outside Victoria, the privacy protection should travel with it.

IPP 10 Sensitive information

The PDP Act places special restrictions on the collection of sensitive information. This includes racial or ethnic origin, political opinions or membership of political associations, religious or philosophical beliefs, membership of professional or trade associations or trade unions, sexual preferences or practices, and criminal record. Organisations can only collect sensitive information under certain circumstances.

Extract from Information Privacy Principles, Office of the Australian Information Commissioner website – www.oaic.gov.au CC BY 3.0 Au



RESEARCH

Smart home devices

Smart home devices such as internet-connected speakers, televisions, fridges and microwaves are designed to make our busy lives easier. Who would not want to be able to control their home with their mobile phone from anywhere in the world? You could turn lights on and off, get your fridge to order the shopping or turn on the heating to make sure your house is warm when you arrive home.



Jarek Kita/Shutterstock.com

FIGURE 6.1 In a smart home, many appliances can be controlled using a smartphone.

Gaming consoles, for example, are found in millions of homes in Australia. These devices rely on internet connectivity to stream video and voice chat. They rely on apps to keep the system and applications up to date. The trouble is that cybercriminals are finding that these devices are a new source of data and information they can use to rob you of money or your identity.

- 1 Review the information presented in Table 6.2. How does this information relate to the use of always-on smart devices in the home?
- 2 In the light of these concerns, what steps can you take to ensure that your privacy is protected?

Health Records Act 2001

The **Health Records Act 2001** is a Victorian state law containing conditions and rules related to how and when an individual's personal health data and information can be collected by both Victorian government agencies and private organisations that either offer health services or handle health records.

This includes health service providers, private organisations, Victorian and local government departments, public hospitals, other public bodies such as Victoria Police and VicRoads, and sole practitioners and partnerships such as doctors, dentists and physiotherapists.

The Health Records Act is based on 11 Health Privacy Principles (HPPs), which, although different, are very similar to the Information Privacy Principles (IPPs). For the purposes of Applied Computing, HPPs 1, 2 and 5 are the focus. These are marked with an asterisk in Table 6.3.

TABLE 6.3 A summary of the Health Privacy Principles. The principles in full can be found in the Act.

***HPP 1 Collection**

Only collect health information if necessary for the performance of a function or activity, and with consent (unless consent cannot be obtained and all other criteria under HPP1 are satisfied). Notify individuals about what you do with the information and that they can gain access to it.

***HPP 2 Use and disclosure**

Only use or disclose health information for the primary purpose for which it was collected or a directly related secondary purpose the person would reasonably expect. Otherwise, you generally need consent.

HPP 3 Data quality

Take reasonable steps to ensure that health information you hold is accurate, complete, up to date and relevant to the functions you perform.

HPP 4 Data security and retention

Safeguard the health information you hold against misuse, loss, unauthorised access and modification. Only destroy or delete health information in accordance with criteria of HPP4.

***HPP 5 Openness**

Document clearly expressed policies on your management of health information and make this statement available to anyone who asks for it.

HPP 6 Access and correction

Individuals have a right to seek access to health information held about them in the private sector, and to correct it if it is inaccurate, incomplete, misleading or not up to date.

HPP 7 Identifiers

Only assign a number to identify a person if the assignment is reasonably necessary to carry out your functions efficiently.

HPP 8 Anonymity

Give individuals the option of not identifying themselves when entering transactions with organisations, where this is lawful and practicable.

HPP 9 Transborder data flows

Only transfer health information outside Victoria if the organisation receiving it is subject to laws substantially similar to the HPPs.

HPP 10 Transfer/closure of practice health service provider

If you are a health service provider, and your business or practice is being sold, transferred or closed down, without you continuing to provide services, you must give notice of the transfer or closure to past service users.

HPP 11 Making information available to another health service provider

If you are a health service provider, you must make health information relating to an individual available to another health service provider if requested by the individual.



ISSUE

My Health Record

The My Health Record system started in Australia in 2018. It provides a fast online summary of your key health issues so any doctor anywhere can access your patient information. For example, you might live in Hamilton, but could be on holiday at Lakes Entrance and be unfortunate enough to be nipped by a crab. When you are taken to the hospital, the treating doctor in Lakes Entrance can access your health record to see that you are allergic to a variety of antibiotics. This will inform the doctor on the best way to treat you.

This is the scenario for all people who chose not to opt out of the system. People who opted out did so mainly because of concerns about data security. Since the Cambridge Analytica scandal in 2018, in which it was revealed that the personal data of millions of people was harvested from their Facebook accounts and used for political purposes, people have been wary of large online personal data repositories.

The Australian Government has established a Digital Health Cyber Security Centre to secure digital national health records across Australia. This centre will monitor and assess any cyber security threat and make regular reviews and improvements to its approach.

How does My Health Record align with the Health Privacy Principles summarised in Table 6.3?

Copyright Act 1968

The **Copyright Act 1968** is an Australian federal law containing conditions and rules related to copyright. **Copyright** gives the owner of an original work (such as a software application, a movie, a song or a photographic image) the right to control who can use the original work, and how.

Copyright applies to any original work created in Australia and is automatic and free. This means that the owner of the original work does not need to apply for copyright; rather, copyright is granted automatically as soon as the work is created. No fee needs to be paid in order for copyright to apply. The only requirement is that the work is in a tangible format, not just a thought in your head.

The IoT poses some problems for the concept of copyright. Internet of Things devices can be used in an individual's home, in their car, in their workplace or when they are shopping, to help them with a range of tasks. Devices on the IoT are interoperable. **Interoperability** is the ability of a device to connect and interact with other devices, particularly devices belonging to different information systems.

The functionality of interoperable devices can be seen to conflict with the purposes of the Copyright Act. For two devices to communicate, they must be able to share software instructions with each other. Software is an original work, and hence is governed by the Act. The owner of the original work has the right to control who can use that work, and in what way.

Australia's Artificial Intelligence (AI) Ethics Principles

Australia's approach to fostering responsible innovation in artificial intelligence (AI) is encapsulated in its AI Ethics Principles. These principles aim to guide businesses, governments and other organisations in designing, developing and deploying AI systems that are user-centric, socially beneficial and accountable. These voluntary principles reflect the growing awareness of how important ethical considerations are with the advancement of AI.

The AI Ethics Principles are as follows.

Human, social and environmental wellbeing

AI systems should be designed and operated to enhance individual, community and environmental wellbeing. This includes ensuring that AI technologies are inclusive and accessible and that they do not harm humans or the environment.

Human-centred values

The development and use of AI systems should respect human rights, diversity and the autonomy of individuals. This principle emphasises the importance of ensuring that AI technologies support democratic values and are aligned with the expectations and norms of the society in which they are deployed.

Fairness

AI systems should be inclusive and accessible and should not involve or result in unfair discrimination against individuals, communities or groups. This involves taking steps to avoid bias in AI algorithms and ensuring that AI applications do not perpetuate existing social inequalities.

Privacy protection and security

AI systems must be secure and respect privacy. Measures should be taken to ensure that personal data is protected and that the integrity of AI systems is maintained to prevent unauthorised access and potential harm.

Reliability and safety

AI systems should be reliable and safe in their operation. They should function accurately and their operation should be predictable to a reasonable extent to avoid unintended consequences.

Transparency and explainability

There should be transparency and responsible disclosure around AI systems to ensure that people understand AI-based outcomes and can challenge them. This principle supports the idea that individuals should be able to understand how decisions are made by AI systems and on what basis.

Contestability

Contestability means that there should be ways for people to question and seek corrections if they think an AI system has made a wrong decision. This principle ensures that there are steps people can take to have decisions made by AI reviewed or challenged if they believe there has been a mistake or injustice. This provides for a process for fixing errors or addressing concerns when AI makes decisions that affect people.

Accountability

Individuals in charge of various stages in the life of an AI system should be clearly recognised and held responsible for how the AI system performs. Additionally, there should be a way for humans to oversee and monitor the AI system's operations.

Read more about Australia's eight Artificial Intelligence (AI) Ethics Principles via the [weblink](#).



Weblink
AI Ethics Principles

Emerging technologies and ethical issues

Ethics are beliefs, principles or standards that individuals, organisations and society have regarding acceptable behaviour. Ethics are sometimes referred to as knowing right from wrong. An **ethical issue** arises when current behaviours or practices could be considered to be the wrong thing to do. These behaviours or practices may not necessarily breach any existing laws, but they still can be considered unethical.

Issues arising from development of emerging technologies

The development of emerging technologies brings significant advancements and conveniences but also raises several ethical concerns that society must address. Issues such as cyber security threats, privacy implications of biometric data collection, and job displacement due to automation and artificial intelligence (AI) are at the forefront of these concerns.

Cyber security threats

Emerging technologies, while offering enhanced capabilities in data processing, communication and automation, also introduce new vulnerabilities and avenues for cyber attacks. These threats include hacking, data breaches, identity theft and the exploitation of personal information. As technology becomes increasingly integrated into daily life, the potential impact of these threats on individuals and organisations grows. Ethical concerns arise regarding the responsibility of developers and companies to ensure the security of their technologies and protect users from harm. It is important to balance the implementation of security measures with users' rights to privacy and freedom.

Biometric systems collecting and storing data

Biometric systems use unique physical or behavioural characteristics for identification and access control, such as fingerprints, facial recognition and voice patterns. While these systems offer a high level of security and convenience, they also raise significant privacy and ethical concerns. The collection and storage of biometric data pose risks of unauthorised access and misuse. There are questions about consent, the extent to which individuals are informed about how their data will be used and the security measures in place to protect sensitive information. Furthermore, there is the issue of permanence; unlike passwords, biometric data cannot be changed if compromised.

Job displacement

The automation of tasks previously performed by humans through technologies such as AI, robotics and machine learning leads to concerns about **job displacement**. While technological advancement can create new job opportunities, there is also the risk of significant job loss in sectors highly susceptible to automation. This displacement can lead to economic inequality, social unrest and challenges in retraining and reemploying the affected workforce. Ethical considerations include the responsibility of technology developers and employing organisations to mitigate these impacts, such as investing in employee training for new roles and contributing to social safety nets.

Issues arising from artificial intelligence (AI)

The use of artificial intelligence (AI) brings about transformative changes across various sectors, enhancing efficiency, personalisation and decision-making processes. However, its widespread

adoption also raises significant ethical concerns that need careful consideration. Among these are: the risks of AI generating biased, discriminatory or harmful content; creating content that could be leveraged in cyber attacks; and issues surrounding copyright infringement.

Biased, discriminatory or harmful content

AI systems, including machine learning models, rely on large data sets for training. If these data sets contain biased or unrepresentative data, the AI is likely to perpetuate or even exacerbate these biases in its output. This can lead to discriminatory practices or harmful content, particularly in sensitive areas such as recruitment, law enforcement, credit ratings and personalised content delivery. Ethical issues include the fairness and transparency of AI systems, the need for mechanisms to detect and mitigate bias and ensuring that AI applications do not reinforce societal inequalities.

How to address this:

- ensure that data sets are diverse, representative and free from biases to prevent discriminatory outcomes
- regularly conduct bias and fairness audits of AI systems to identify and address emerging biases. These audits should be carried out by independent experts or third parties to ensure objectivity.

Content and the potential for cyber attacks

AI can be employed to generate sophisticated phishing emails, create malware or automate hacking attempts, making cyber attacks more efficient and harder to detect. The dual-use nature of AI technologies, where the same technology can have beneficial uses as well as potential for harm, presents ethical dilemmas about the development, distribution and control of these technologies. There is a pressing need for ethical guidelines and security measures that prevent the misuse of AI in cyber attacks while promoting its positive applications.

How to address this:

- assess potential forms of misuse and implement safeguards during the early stages of development
- deploy systems capable of monitoring and detecting malicious AI-generated content in real time, using AI-driven anomaly detection techniques.

Problems with copyright

AI's capability to generate text, images, music and other forms of content that closely resemble human-created works raises copyright concerns. The use of existing copyrighted materials to train AI models without proper licensing or attribution can lead to disputes over the ownership and copyright of AI-generated content. This situation challenges existing legal frameworks and raises questions about originality, authorship and the rights of creators whose works contribute to training AI models. Ethical considerations include respecting intellectual property rights and developing fair compensation models for creators whose works are used in training AI.

How to address this:

- develop and use digital watermarking techniques to embed invisible markers in AI-generated content
- encourage the use of clear, transparent licensing agreements that define the use of copyrighted material for training AI models.

**CASE STUDY****DataGenix**

A company named DataGenix, co-founded by Isaac, Keira and Omi, has developed an advanced artificial intelligence (AI) system designed to revolutionise content creation across various industries. The AI, named CreatoAI, is capable of generating text, images and videos to assist companies in marketing, education and entertainment. However, as CreatoAI's capabilities grow, so do the ethical concerns surrounding its use.

DataGenix's CreatoAI is initially celebrated for its efficiency and creativity, quickly becoming a favourite tool for content creators worldwide. Keira, with a background in software engineering, focuses on refining CreatoAI's algorithms. Isaac, an expert in digital marketing, handles client relations and explores market needs. Omi, the legal adviser, ensures that the startup's operations comply with existing laws and regulations.

The problem

As CreatoAI gains popularity, it begins to show unintended biases in its content generation. A report highlights that the AI consistently portrays certain demographic groups in stereotypical roles while offering more diverse and positive representations to others. This issue arises from the AI's training data, which Keira realises is not as diverse or balanced as it should be. The biased output sparks a public backlash, forcing the DataGenix team to confront the ethical implications of their oversight.

Meanwhile, an anonymous group starts using CreatoAI to create highly convincing fake news articles, leveraging the AI's ability to generate persuasive and authoritative content. Isaac discovers this misuse after encountering a sophisticated scam mimicking a campaign the team had developed for a client. This incident exposes the potential for CreatoAI's technology to be used for cyber attacks, raising ethical questions about the responsibility of AI developers to prevent such misuse.

Finally, Omi receives a legal letter from a renowned author, claiming that CreatoAI has generated a series of short stories closely resembling the author's unique style and themes, which were included in the AI's training data set without prior consent or proper authorisation. This copyright dispute highlights the complexities of using existing copyrighted materials to train AI systems and the need for clear guidelines and permissions.

Addressing the problem

Faced with these ethical dilemmas, the DataGenix team takes several actions:

- Keira leads an initiative to audit and diversify the training data sets. DataGenix collaborates with external experts to develop algorithms that can identify and mitigate bias in AI-generated content.
- Isaac initiates partnerships with cyber security firms to develop protocols that prevent the misuse of CreatoAI for malicious purposes. DataGenix also releases guidelines for users, emphasising ethical usage and the legal consequences of misuse.
- Omi works on establishing clearer policies for content generation, including obtaining licences for training materials and developing a system to ensure that AI-generated content does not infringe copyright laws. DataGenix also explores creating a content attribution feature to acknowledge the sources of inspiration for AI-generated works.

The DataGenix team recognise that addressing these ethical issues is an ongoing process, requiring continuous vigilance, improvement and dialogue with stakeholders. By taking proactive steps, DataGenix aims to set a new standard for ethical AI development, ensuring that CreatoAI benefits society while minimising harm and respecting legal and ethical boundaries.

Documenting and monitoring progress

A **project** consists of a series of interrelated tasks that need to be completed to be able to create a **solution** to a **need** or **opportunity**.

A project has the following characteristics:

- a clearly defined purpose
- a start time
- a limited timeline
- a number of tasks.

Examples of a project include:

- building a house
- writing an essay for school
- moving house
- installing a new computer network
- developing a software solution.

You can draw on a number of tools to assist you when planning projects. Most are designed to show different aspects of a project's status, such as the time taken or which tasks depend on one another. The project-management tools discussed below should be used to complement one another, not in isolation. Project-management software does not create tasks and assign resources. You must make informed decisions, and the software will help you with the management, documentation, presentation and communication of project information. You can update the documentation easily when changes are made during the course of the project.

For Unit 2, Outcome 1, you will, in collaboration with other students, analyse, design, develop and evaluate an **innovative solution** to an identified need or opportunity involving a digital system. The first part of that task will involve creating a project-management plan, then monitoring the project and updating the plan as required during the course of the project. The first part of the project-management plan will be the project table; this is followed by a Gantt chart. Finally, keep a record of the progress and any changes required to the timeline in the project journal. These project-management tools are all explained below.

Project table

The project manager will brainstorm, on a separate sheet of paper or word-processing document, all the possible tasks involved in analysing a problem, and designing, developing and evaluating a solution (Table 6.4).

TABLE 6.4 The main headings of a project table for publishing a report, with sample data

Task number	Task name	Duration (days)	Resources required	Task milestone (Y/N)	Dependent tasks (predecessors)
1	Departmental reports	5		No	N/A
2	CEO's report	3		No	N/A
3	Proofread	2		Yes	1, 2
4	Obtain artwork	5		No	N/A
5	Cover design	3		No	4

Gantt chart

A **Gantt chart** provides a standard format for displaying project schedule information. It lists the project tasks worked out in the project table, and their corresponding start and finish dates, in a calendar format (Figure 6.2). Gantt charts not only show a timeline for completion of the project, but they can also highlight tasks that are critical to the timely completion of a project. Using a Gantt chart makes it easy for the project team members to see when tasks need to start and how long they should take. Many Gantt charts also show milestones and basic task dependencies. **Dependencies** show the relationships between tasks, when a task should begin or end in relation to another task. Although you can use dedicated project-management software to generate Gantt charts, simple ones can be constructed in spreadsheet programs.

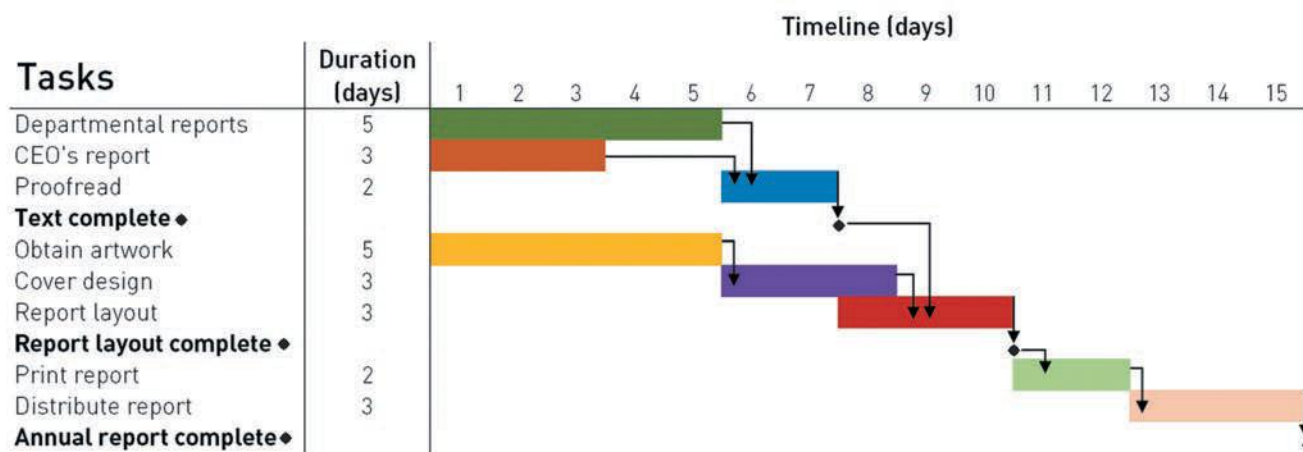


FIGURE 6.2 An example of a Gantt chart created in Microsoft Excel. It displays the information shown in the project table (Table 6.4). Simple dependencies have been shown with arrows. Different colours can be used to make it easier to identify individual tasks.

Annotations

Annotations are comments or notes that are placed on the Gantt chart to explain any changes that may be made. These provide useful information to project team members about problems that may be occurring and may identify any recurring issues with the project. These annotations may also become useful after the project is completed, providing lessons about things that should be avoided in future projects.

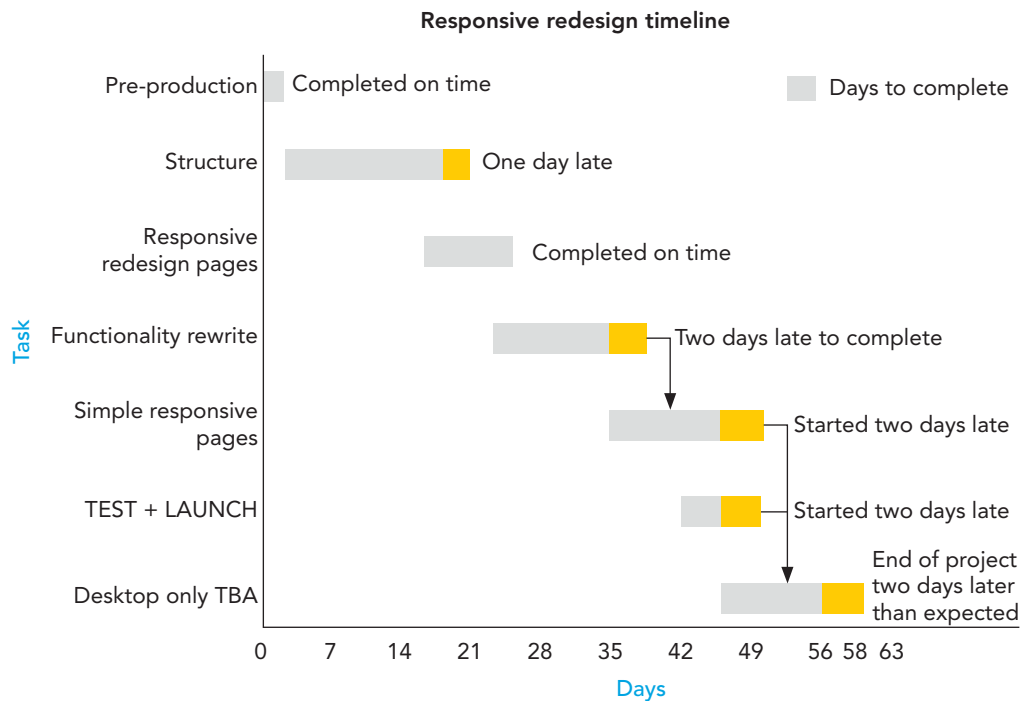


FIGURE 6.3 Gantt chart with annotations

Adjustments

Adjustments are any changes that may need to be made to the timeline as a result of events that have occurred as the project progressed. Sometimes, if a task takes longer to complete than expected, the delay may not have any effect on other tasks. If this situation occurs, adjustments to the project plan will be required.

			Time/ Date	M	T	W	T	F	S	S	M
No	Task	Dur	Pred	23	24	25	26				
A	Design tickets	1									
B	Design website	3									
C	Approval	1									
D	Design completed	0									
E	Create ticket	2									

FIGURE 6.4 Gantt chart before adjustments

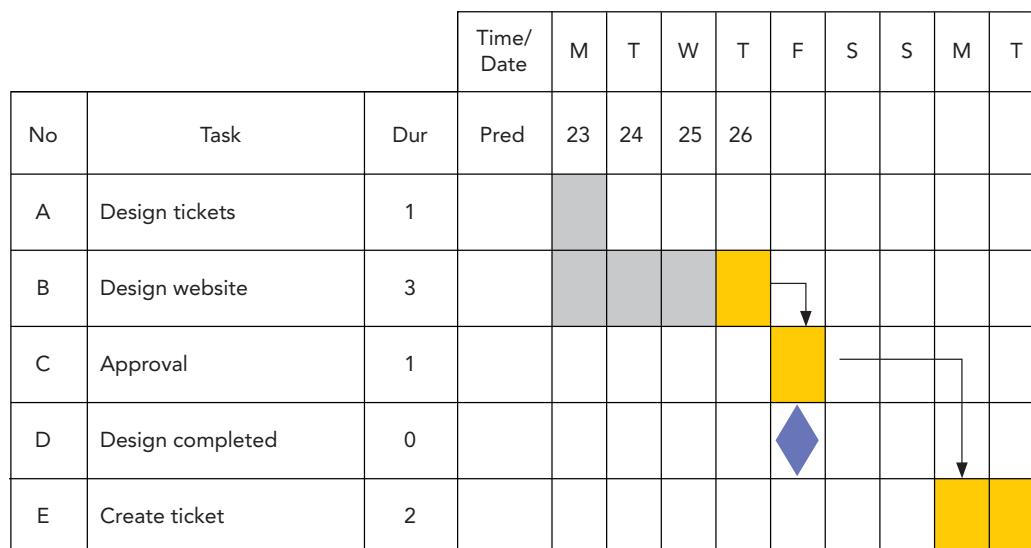


FIGURE 6.5 Gantt chart after adjustments

Project journals

Once a project is under way, a **project journal** can be used to document the development of the solution. A project journal is similar to a diary in which events that affect each task in the project are written down to create a history of the project. These journal entries may assist by providing details that can be used to complete tasks, or by indicating problems that occurred so they can be avoided in later stages of the project.

	A	B	C	D	E	F	G	H	I	J
1	Issue #	Date	Reported by	Item/Functionality	Description	Issue type	Priority	Impact	Suggested fix	Status
2	1	2025-05-09	Developer	Input box submit button	Spacing too close to text box	Usability	Medium	High	Increase spacing. Introduce colour and label with meaning	Resolved, adjusted size, colour and labelling
3	2	2025-05-09	Developer	Title	The font size of the main input screen is too small, hard to read	Design	High	High	Adjust the size and colour contrast	Resolved, larger font used, darker colour
4	3	2025-05-09	Developer	What to do is not clear	After reading the introduction screen, the search and sort buttons are not clearly marked.	Usability	Medium	Medium	Adjust the size and colour shading of the button. Add a label	Partly resolved, need to work out how to include a label inside
5	4	2025-05-09	Developer	Image quality of LOGO	Image is very poor quality	Data	Medium	Low	Locate better original and re-capture better quality screenshot. Completely redraw to create new original.	Still looking to locate image, yet to be attempted
6	5	2025-05-10	Reviewer	Sources and Privacy statement	There is no acknowledgement of sources, destination of data, or privacy statement	Data	Low	Low	Add sources citation link. Add link to Privacy information and Terms and conditions to the lower left of the opening screen.	Yet to be completed

Microsoft Corporation

FIGURE 6.6 Project journal template created in Microsoft Excel. Columns C, F, G and H are data validation drop-down lists. Columns G, H and J also have conditional formatting applied to indicate priority and impact by colour: low, medium, high, critical and showstopper. This template is available on Nelson MindTap.



Resource
Project journal template

Version control

Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later. It is an essential tool in the development of solutions, particularly in software development, and its usefulness also extends to any collaborative project involving digital documents. Version control serves important functions in the documentation of development, task delegation and monitoring of project progress, making it indispensable for managing complex projects.

Version control systems provide a comprehensive history of the changes made to a project's files. This history includes details about what changes were made, who made these changes and when they were made, allowing teams to track the evolution of their project over time. This is particularly useful for understanding the reasoning behind certain decisions and troubleshooting issues as well as analysing the development process. By documenting each step of the solution's development, teams can easily revert to previous versions if a new change causes problems, ensuring the integrity and continuity of the project.

In collaborative projects, **task delegation** is streamlined with version control systems. Version control tools like Git allow the use of branches, which are separate copies of the project that can be worked on independently by different team members. This feature enables simultaneous work on different parts of the project without interference, as each branch can represent a specific task or feature assigned to a team member. Upon completion, these branches can be merged back into the main project, integrating the work seamlessly. This process ensures that task delegation is clear, organised and efficient, with minimal conflicts between the concurrent development efforts of team members.

Version control systems are also invaluable for monitoring the progress of a project. By examining the record of all changes made (the commit history), project managers and team members can judge the project's advancement towards its goals. Each commit acts as a checkpoint, indicating what has been accomplished up to that point. This visibility allows project managers to identify bottlenecks, assess individual contributions and ensure that the project is on track. Furthermore, version control supports project planning and review processes by providing a clear, time-stamped log of the project's development, facilitating the assessment of progress against timelines and milestones.

Collecting data

As part of the analysis stage of the problem-solving methodology, you will identify the requirements, constraints and scope of the solution. To do this, you will need to collect data about the identified need or opportunity.

Techniques that can be used to collect data, such as observation, **interviews** and **surveys**, are discussed in Chapter 1. Refer to pages 6–8 to review these techniques.

A need responds to an existing problem (reactive), whereas an opportunity is the development of a new solution for a problem that doesn't necessarily already exist (proactive).



CASE STUDY

An artistic endeavour at Keen College: Identifying the opportunity

Ms Curry is the Year 11 Art teacher at Keen College. Her students are currently working on a major piece of assessment, in which they are experimenting with different materials and techniques to produce a piece of artwork that is significant to them. Each student must keep a visual diary to document their artistic journey. Ms Curry is very proud of her students' achievements and wants to make their finished artworks available to the whole school community, including other students, teachers and all parents.

Ms Curry mentioned this idea to Mrs Mustard, the Year 11 Applied Computing teacher. Ms Curry said that she thought that perhaps a digital solution would work best. Mrs Mustard was delighted as she has been looking out for a possible project for her students. Ms Curry and Mrs Mustard have decided to combine the talents of their two classes to come up with a solution to this opportunity.



Layland Masuda / Shutterstock.com

FIGURE 6.7 Ms Curry working with one of her art students

Characteristics of creative and innovative solutions

Creative and innovative solutions stand out due to several key characteristics that distinguish them from more conventional approaches. These characteristics not only define the nature of such solutions but also contribute to their success and impact. In Applied Computing, you should be considering the originality and effectiveness of the solution, as well as how user-centred it is, and the level of disruptiveness it might cause.

Originality

Creative and innovative solutions are often groundbreaking, novel and/or unique, offering new perspectives or approaches to solving problems. This doesn't necessarily mean that every aspect of the solution must be unprecedented, but the combination of ideas, application or problem-solving method should present something that is not merely an imitation of existing solutions. Original solutions can inspire further innovation and set new benchmarks in their respective fields.

Effectiveness

Effectiveness measures how well a solution addresses the problem it was designed to solve. An innovative solution must not only be novel but also functional and practical. This means it successfully meets the needs or solves the problems of its target users or beneficiaries. The effectiveness of a solution is a critical indicator of its value and sustainability over time.

User-centred

Being user-centred means that the design and development of the solution are highly informed by the needs, preferences and behaviours of its intended users. This approach involves a thorough understanding and empathy of the user's context, challenges and desires. User-centred solutions are tailored to provide a positive and intuitive user experience, making them more likely to be adopted and praised by their target audience. This characteristic ensures that the solution is not only innovative but also meaningful and relevant to the people it aims to serve.

Disruptiveness

Disruptiveness indicates the extent to which a solution challenges and changes traditional ways of thinking and doing things. Disruptive solutions often redefine the boundaries of what is possible, introducing shifts in behaviour, industry standards or market structures – consider, for example, the recent disruptiveness of advanced language model systems such as GPT. These disruptive solutions can create new markets or significantly alter existing ones. While disruption can be seen as a marker of significant innovation, it also implies the solution's capacity to cause shifts in power dynamics, leading to the obsolescence of older technologies, products or methodologies.

Solution specifications

The problem-solving methodology is used to create a solution that will meet the needs of the users.

The first stage of the problem-solving methodology is the analysis stage. This is often considered the most crucial stage. Many organisations invest a lot of time, effort and money in this stage. They consider that getting the analysis stage correct and having a clear picture of what the user requires will save time and money in the latter stages.

The analysis stage is typically about 'What?' questions. What is the current information need? What is required to meet the current information need? What constraints may restrict the requirements? What is the scope of the solution?

The analysis stage consists of three activities:

- determining solution requirements
- determining solution constraints
- deciding on the scope of the solution.

Solution requirements

Solution requirements state what the client needs from the solution. What features do they want in the solution? Solution requirements can be broken down into functional and non-functional requirements. These were covered in Chapter 3, pages 114–17.

6.2 THINK ABOUT APPLIED COMPUTING

Explain how completing a thorough analysis may save time and money in the latter stages of the methodology.

Functional requirements

Functional requirements are directly related to what the solution will do. These become the functions that the software incorporates. Examples of functional requirements for a smart refrigerator might include the ability to:

- detect items stored in the refrigerator
- track use-by and expiry dates
- control temperature by drawer or compartment
- send alerts when issues are encountered with the refrigerator
- control settings using a smartphone app.

Non-functional requirements

Non-functional requirements are the characteristics users or clients would like the solution to have but that do not affect what the solution does. Examples include that it:

- is user-friendly
- has a clear user interface
- does not display any personal details
- is compatible with different operating systems
- has app portability so it can operate on devices of different sizes.



CASE STUDY

An artistic endeavour at Keen College: Project planning and solution requirements

Ms Curry and Mrs Mustard arranged to get their Art and Applied Computing (AC) students together one lunchtime. There were three students who took both classes, so these students were designated as the 'go-betweens', taking on the role of keeping both classes up to date with what was happening.

The AC students want to apply the problem-solving methodology to finding a solution to this opportunity. The students sat down together and generated a project table of the key tasks that would have to be performed. Then they created a Gantt chart, which they shared with the students of both classes so that everyone was aware of the tasks, their dependencies and the timeline they needed to work within.

The students would then document all stages of the process in the project journal (see page 220) and update the Gantt chart if task requirements changed.

The first stage of the problem-solving methodology is *analysis*, starting with determining the solution requirements. The AC students decided to interview each art student individually to find out what they wanted. Half the AC students agreed on the interview questions and then conducted the interviews.

The other half of the AC class created a Survey Monkey survey about user requirements and sent this out to all teacher, student and parent email addresses stored on the school database.

As a result of the interviews and the survey, it was determined that the solution requirements would be as follows.

Functional

- Display one piece of art from each art student.
- Present, in some format, each art student's visual diary.

- Make the gallery available to all students, teachers and parents.
- Have the ability to add more art pieces.
- Alert users when more art pieces have been added.

Non-functional

- Image files need to be downloaded in a reasonable time.
- The solution has to be accessible.
- Solution is available to use on all devices.
- The gallery is available 24/7.

Solution constraints

Solution constraints are factors that may limit or restrict the solution requirements, both functional and non-functional.

Like requirements, constraints can be broken down into groups: economic, technical, social, legal and usability. Constraints were covered in Chapter 3, pages 117–19.

6.3 THINK ABOUT APPLIED COMPUTING

List three technical constraints that developers of a drone may need to consider when developing the product.

An artistic endeavour at Keen College: Constraints

On the basis of the solution requirements identified from the interviews with the art students, the AC students brainstormed all the constraints that they could think of and categorised these under headings, as follows.

Economic

- School Council has made \$500 available to produce the solution.
- Ms Curry wants the solution available one week before the school Open Day at the start of Term 4.

Technical

- The school has a Bring Your Own Device program, but most students use a Windows laptop.
- Download speeds at the school can be very slow.

Legal

- Student privacy if student names are published
- Copyright of student work

Usability

- Accessibility for deaf and blind members of the school community

CASE STUDY



Scope of the solution

After the solution constraints are identified and the solution requirements are adjusted based on the constraints, the scope of the solution can be created.

The **scope of the solution** outlines the boundaries or parameters of the solution so all stakeholders are aware of exactly what the solution will contain and what it will not contain. It is best to outline at the start of the project what will and will not be included in the solution. This can prevent disagreements later in the project between the user and the developer. Scope was covered in Chapter 3, pages 113–14.

6.4 THINK ABOUT APPLIED COMPUTING

List three problems that a clear scope of solution can avoid later in the project.



CASE STUDY

An artistic endeavour at Keen College: Scope of the solution

The AC students considered the solution requirements on the basis of the identified constraints. They decided that, as the time and funds available for the production of the solution were limited, they would remove the students' visual diaries from the functional requirements. This would then enable them to produce the solution within the time frame required and the budget available. They agreed that they would publish each artist's name along with a description of the artwork. A copyright notice would be visible on the same page as each of the artworks.

They agreed that the solution scope would be as follows.

In scope

- Display one piece of art from each art student.
- Make the gallery available to all students, teachers and parents.
- Thumbnail image files will be viewed on screen without the need to download the full-size image.
- Each piece of artwork will have the artist's name and a short description.
- There will be a visible copyright notice on each page displaying artwork.

Out of scope

- Have the ability to add more art pieces.
- Alert users when more art pieces have been added.
- Solution is available to use on all devices.
- Have the gallery available 24/7.
- Present, in some format, each art student's visual diary.

Techniques for developing innovative solutions

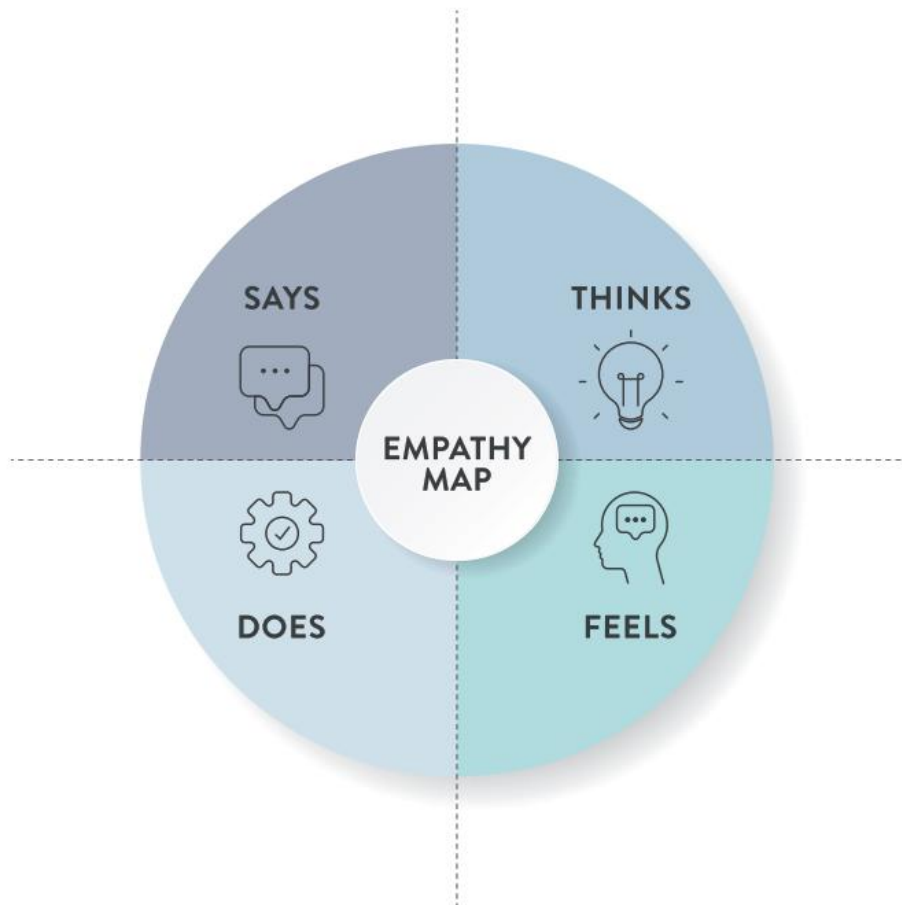
Empathising with the end user and ideating to explore multiple perspectives are important techniques when developing innovative solutions. These approaches are grounded in the design thinking process, which is a human-centred methodology to problem solving. They help in creating more effective, user-friendly and innovative products or solutions.

Empathy is the foundation of the design thinking process. It involves deeply understanding the experiences, challenges, motivations and needs of the people you are designing for. This step goes beyond mere observation or data collection; it is about putting yourself in the users' shoes to gain a genuine insight into their lives and perspectives. This understanding is critical for defining the problem accurately and coming up with solutions that truly address users' needs.

Empathy mapping

Empathy mapping is a collaborative tool used to gain a deeper understanding of a target audience's experiences, behaviours and feelings. It helps teams visualise and articulate what they know about a particular user or customer. An empathy map is typically divided into four quadrants, each representing a different aspect of the user's experience.

- 1 **Says:** This quadrant captures what the user says out loud in an interview or during usability testing. These are direct quotes or statements about their thoughts, feelings or experiences.
- 2 **Thinks:** This section delves into what the user is thinking throughout the experience. While it often overlaps with what they say, it can also include thoughts they might not articulate.



Whale Design/Shutterstock.com

FIGURE 6.8 An empathy map

- 3** Does: This quadrant records the user's actions and behaviours. What do they do in response to the experience? How do they interact with a product or service?
- 4** Feels: This part captures the user's emotional state, often inferred from their statements, behaviours and body language. It seeks to understand the user's emotional journey.

By filling out an empathy map, teams can create a shared understanding of the user, leading to more user-centered designs and solutions.

Designing educational software for enhanced learning

An educational software development project is being designed to enhance the learning experience for secondary school students. Kestrel is a software developer and Timur is a secondary school teacher who faces challenges engaging his students in software development and applied computing subjects.

Kestrel is tasked with developing the software that will help secondary school students be more engaged. Recognising the importance of empathising with the end users (in this case, both students and teachers), Kestrel decides to focus first on understanding the challenges and needs from the teacher's perspective.

Kestrel begins by arranging a series of meetings and observation sessions with Timur to gain a deeper understanding of the challenges he faces in the classroom. From these meetings and observations, Kestrel creates an empathy map (Figure 6.9).

CASE STUDY





'My students struggle with understanding abstract concepts in software development.'

'I wish I had more resources to make lessons more practical and engaging.'

'Traditional teaching methods are not keeping the students engaged.'

'I want my students to reach their full potential but feel limited by the current curriculum.'

SAYS

DOES

Incorporates as many practical examples as possible within the limitations.

Looks for new methods and tools to engage students.

Spends extra time preparing lessons to make them more interesting.

Observes students' reactions and adjusts teaching methods accordingly.

'How can I make software development more interesting for my students?'

'I need more practical tools and resources to help my students learn better.'

'There must be a better way to teach these applied computing concepts effectively.'

'I feel frustrated when my students don't understand or lose interest.'

THINKS

FEELS

Passionate about teaching and helping students succeed.

Frustrated by the lack of resources and curriculum constraints.

Motivated to find new ways to enhance student engagement.

Disappointed when students do not grasp concepts or lose interest.



FIGURE 6.9 Kestrel's empathy map

Through his observations, Kestrel has learned that Timur's students often struggle with abstract concepts in software development and find it hard to stay engaged with traditional teaching methods. Timur also expresses his desire to incorporate more practical, hands-on learning experiences but is limited by the current curriculum and resources available.

As shown in the empathy map, Kestrel has identified Timur's passion for teaching and his frustration when students do not reach their full potential due to these limitations. He empathises with Timur's situation and becomes more motivated to develop a solution that can make a real difference.

Armed with these insights, Kestrel returns to his team to brainstorm ideas that can address the challenges identified.

Ideating for exploring multiple perspectives

Ideation is the process of generating a broad set of ideas, free from judgement, with the aim of identifying innovative solutions. This stage leverages creativity and diverse thinking to explore a wide range of possibilities.

Exploring multiple perspectives in ideation involves leveraging various creative and structured techniques to generate a wide range of ideas and solutions for a problem. Each technique brings a unique approach to thinking and can help uncover innovative solutions that might not be immediately apparent. By incorporating techniques such as brainstorming, mind mapping, substitution, combination, adaptation, stripping down to basics, comparison, 'sleeping on it', research and visualisation, teams can ensure a rich and diverse ideation process.

Generating possible design ideas

Brainstorming

A common technique is **brainstorming**. Often in a brainstorming session, the participants are encouraged to offer any possible idea they can think of that may solve the problem, regardless of how crazy or far-fetched the response may be considered by others.

When brainstorming, try not to hamper the imagination by rejecting ideas too soon. Brainstorming is a process in which ideas are presented in a non-judgemental, spontaneous, unstructured – and admittedly somewhat haphazard – process. The only rule is that no idea is criticised or rejected; every idea, no matter how outrageous or silly, goes onto a list of possible solutions.

Participants must have no fear of being ridiculed or judged, making mistakes or breaking rules. While some or many ideas *do* turn out to be ridiculous, sometimes a half-baked, half-comical concept may in fact turn out to be an unexpected work of creative genius, or it may stimulate a related idea that would be perfect.

There are certain rules that you need to adhere to when you are running a brainstorming session. The most important one is that no one judges any contribution. You must accept all contributions. An idea that may seem slightly crazy to begin with can sometimes be workshopped into a great idea. In the 1970s, a brainstorming session came up with the idea of a pet rock. The idea was workshopped, and before long you could buy not only a pet rock but a pet rock house and a training manual. Everyone had to have a pet rock in the 1970s and the idea made the company millions of dollars. It was the pre-technological version of a Tamagotchi.

Make sure that everyone listens to everyone else's ideas. Specify that only one person talks at a time and there is only one idea at a time. This not only ensures that even the shyest member of the group will contribute, but also makes it easier to record the ideas. Using these rules will assist you in getting a large number of ideas with which you can work.

One strategy behind the use of brainstorming is that allowing participants to be free-thinking means they are not constrained by traditional approaches to solving a problem. The hope is that they may come up with a creative way to meet the need or opportunity.

Mind mapping

Mind mapping is ideal for complementing the process of brainstorming. Mind mapping is a technique for quickly generating and linking ideas. It is a creative and flexible tool that enables you to add, connect, organise and reorganise ideas. Mind-mapping software is generally flexible enough that you will not need to stop very often to learn how it works; in other words, when you are mapping, your creative flow will not often be interrupted.

Unlike old-fashioned sheets of paper or whiteboards, electronic mind maps can stretch endlessly in any direction. You can easily add or remove links between items or allow entire branches of thought to be moved to new locations, and you will not face the laborious task of copying out all of the scribbled ideas at the end of the brainstorming session. The mind map can be saved for later development, printed, saved as an image, or transferred to a word processor.

6.5 THINK ABOUT APPLIED COMPUTING

Other techniques used to generate solution ideas include:

- experimenting with similar solutions
- role-playing
- visualisation
- reverse engineering
- questioning.

Choose one technique and find out what it involves.

FIGURE 6.10 Mind maps allow you to generate and link ideas easily.



Andrey_Popov/Shutterstock.com

Generating alternative ideas

When designing the solution to a problem, the first idea you have is rarely the best one. A different strategy might be cheaper, easier, faster, more effective, or may better meet the client's demands. Questions to be asked when thinking about alternative solutions are:

- What else can we ...?
- What if ...?

Substitute

Replace part of the problem with something else. For example, if you have always keyed contact details into a spreadsheet, you might find it more useful to use a database instead.

Combine

Join unconnected things together, such as making travel easier by combining a suitcase and a scooter to make a scootcase (Figure 6.11).

Adapt

Use an existing component in a different way, such as using mind-mapping software to create a site map for a website. The first spreadsheet was created using the concept of paper-based accounting books.

Strip

Reduce the problem right back to its most basic parts and see what is left. For example, the tiny and cheap computer, the Raspberry Pi, is a stripped-down Linux PC with minimal components.

Inspecting the basics may reveal the nature of a problem more clearly.

Compare

Ask yourself, 'What other thing do I know that resembles *this* problem, and how does that other thing work?' For example, when sending a number of print jobs to a single printer, how can they be handled? Like a group of people waiting at a gate, you could organise them into a queue and process them in the order of their arrival.

Sleep on it

Creators often reach a point where they can make no further progress. Rather than dwelling on the same failed ideas, it is often better to let them go and think of something else. While



WENN Rights Ltd / Alamy Stock Photo

FIGURE 6.11 A scootcase – a suitcase combined with a scooter

the front of your brain is enjoying a walk with your friend, or an episode of *My Kitchen Rules*, the back of your brain will busily be pulling ideas together to create a solution.

Research

Thomas Edison said: 'Through all the years of experimenting and research, I never once made a discovery. I start where the last man left off.' It is important to learn from your predecessors so you do not waste time re-inventing the wheel.

How have other people solved problems similar to the one you face? You are unlikely to be the first person in history to have faced such a problem before. How have others coped? Their successes may lead you in the right direction, and their failures may prevent you wasting time.

Visualisation

Geniuses often make their thoughts visible because words cannot adequately convey the ideas they have. Albert Einstein was famous for his non-verbal thought experiments. He visualised travel at the speed of light as travelling on a train. He said that written words and numbers did not play a significant role in his thinking process.

Selecting the preferred design

Once alternative solutions have been workshopped, the next step is to select the one that will provide the best solution to meet the need or opportunity. While one idea may be attractive to the developer, the client may have non-technical constraints or priorities that will make one strategy more attractive than another.

Providing a range of solution ideas lets you choose the one that best suits your requirements. You may have used an idea successfully in the past, but it may not be appropriate in the current circumstances. Although previously proven strategies can be useful, you need to be willing to think outside the box. Old strategies will not work for you in every situation, and it is lazy and unimaginative to assume that they will.

A successful problem-solver will consider current functional and non-functional requirements and relevant constraints to develop an imaginative range of options, from which the best solution can be chosen and developed into a detailed design.

The criteria for choosing the best solution include:

- how easy it is to use
- accessibility
- how long it will take to implement
- scalability (how easily the product can be increased in capacity)
- its scope for future modification and enhancement
- the degree to which it satisfies all requirements
- the degree to which it copes with constraints
- ease of implementation
- development cost, future running and maintenance costs
- the amount of labour required to create it and keep it working
- the amount of disruption likely to be caused to the organisation
- compatibility with existing hardware, software, data and procedures
- the amount of training required for staff.

Some decisions can be very difficult, and require careful balancing of competing needs – usually, cost against quality. A solution that is cheap and quick to produce may be barely competent, quickly wear out or be unpleasant to use. A superior solution that would lead to a solution with a long life and happy users will probably take longer to produce and cost more.



CASE STUDY

An artistic endeavour at Keen College: Choosing an innovative solution

Now that the AC students have scoped out the solution, they have to come up with the type of solution they are going to use. As they have a class of 21 students, they decide to split the class into three equal groups. One group will use the brainstorming technique to come up with innovative solution ideas, one group will try the mind-mapping technique and the other group will search the internet to see what has been done before to solve a similar opportunity. They think that by doing it this way they will come up with the most ideas.

After all three techniques have been used, they combine their ideas and compare them with the functional and non-functional requirements and the constraints. In this case, they prune down their list to the following five ideas:

- 1 physical gallery with QR codes on each piece of artwork to supply the information
- 2 physical gallery with AR markers on each piece of artwork to supply the information via mobile device app
- 3 physical gallery with Bluetooth beacons that would supply different information depending on the range
- 4 web page gallery
- 5 printed catalogue with AR capability.

The students then compare their cut-down list with the selection criteria to see how each fared (see Table 6.5). Each idea is rated on a scale of 1–10, where 1 is poor and 10 is excellent. The scores for each idea are then added to get a total.

TABLE 6.5 Each idea ranked against criteria

Criteria	Idea 1	Idea 2	Idea 3	Idea 4	Idea 5
How easy it is to use	7	6	8	8	8
Accessibility	4	4	4	4	2
How long it will take to implement	8	6	6	9	5
Scalability	8	9	6	9	5
Future modification and enhancement	9	9	7	9	4
Satisfies all requirements	6	6	6	9	8
Copes with constraints	8	8	8	9	7
Ease of implementation	8	6	6	9	5
Development cost, future running and maintenance costs	9	5	3	8	4
The amount of labour required	8	3	6	8	4
The amount of disruption	8	8	8	7	9
Compatibility with existing hardware, software	6	7	4	9	6
Amount of training required for staff	7	6	5	8	7
Total	96	83	78	106	74

The class decides that a web page gallery (Idea 4) will be the best solution for the users as it will make the artwork available to all users when they want to view it, and it will be ready on time and within budget.

Developing the preferred design

A good solution will undergo a rigorous design stage before any attempt is made to implement it. The solution must address the stated need or opportunity and meet all the functional and non-functional requirements listed in the analysis stage.

Whereas in the analysis stage the focus is on what needs to be done, the design stage focuses on how it will be done. Designing a solution can occur without using a computer at all. Much time can be wasted if a design has not been properly planned before work starts on developing the solution.

In the problem-solving methodology, the two activities involved in the design stage are:

- designing the solution
- creating evaluation criteria.

To design the solution, a range of design tools are used to represent both the appearance and the functionality of the solution. The type of design tools used for a particular solution will vary depending on the format of the solution (website, database, drone, software application, etc.).

Types of design tools include (but are not limited to):

- mock-ups (see Chapter 2)
- pseudocode
- site maps
- storyboards.

Pseudocode

If you have decided to create a programming solution, you would use pseudocode as the design tool to represent the functionality. **Pseudocode** is used to represent the logic or steps of processing that occur when a solution is run or executed. This logic or the steps of processing are often referred to as a solution's algorithm.

Pseudocode is frequently used when developing a solution using a programming language, but can also be used to represent the logic in solutions developed using other software types. Pseudocode combines the structure of programming language code with instructions or steps written in plain English, as it is created to be read by humans, not a computer.

Below is the pseudocode representing the steps of processing that will occur in a solution that a teacher will use to calculate a student's average grade and award based on their test results.

```

ALGORITHM calculateResult()
BEGIN
    INPUT Mark 1
    INPUT Mark 2
    INPUT Mark 3
    Total = Mark 1 + Mark 2 + Mark 3
    PRINT Total
    Average = Total / 3
    PRINT Average
    IF Average >= 50 THEN
        PRINT Pass
    ELSE
        PRINT Fail
    ENDIF
END
  
```

FIGURE 6.12
Pseudocode to calculate
student average grades

Pseudocode is commonly used when creating solutions using a programming language. Advantages of pseudocode are that it allows the logic of the solution to be created before development begins; it allows the algorithm to be bench-tested; and changes can be made much more quickly and inexpensively than would be possible if changes were required to the solution when being developed.

Refer to Chapter 3, pages 124–5 for more information on pseudocode.

Site maps

A site map is used to represent the functionality of a solution. A **site map** shows how the information has been structured within the solution, and how a user can navigate to find the required information. A site map is commonly used as a design tool for websites, but can also be used to show the structure of information presented using other formats, including games, multimedia, interactive solutions and smartphone applications.

Site maps help to display the information architecture of a solution. A site map can also be referred to as a hierarchy or tree diagram.

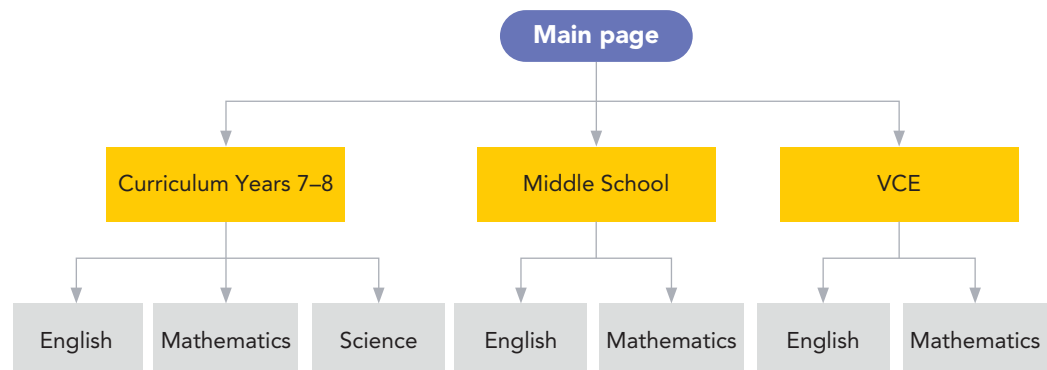


FIGURE 6.13 A site map for a subject selection website

Site maps are useful in that they help to identify how the information should be organised and presented, the number of sections needed within the solution (information architecture) and how a user can access the different sections (navigation).

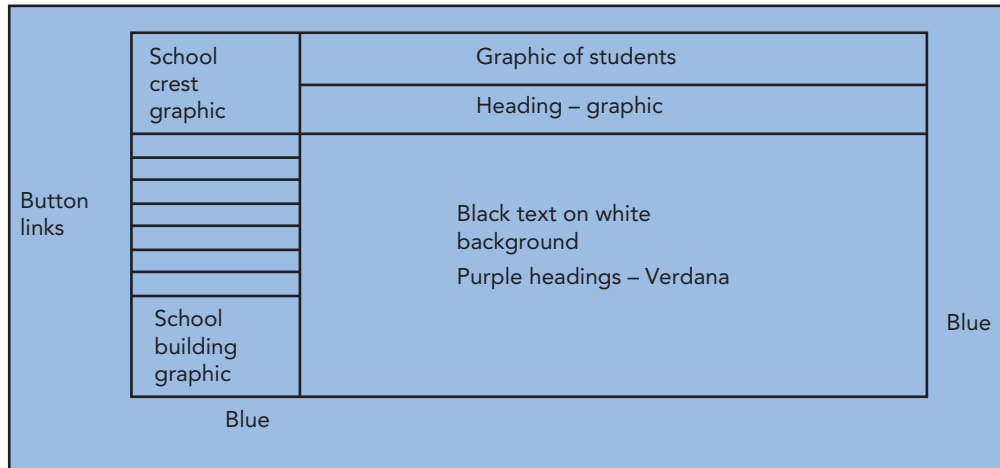
Storyboards

A storyboard is a design tool that can be used to represent the functionality of a solution, but also contains elements about the appearance of a solution.

A **storyboard** shows the features of an individual section of a solution. A series of storyboards can be used to represent the entire solution. Storyboards are commonly used when designing websites, but can be used for other software types.

A storyboard can contain a basic layout diagram showing how the section of the solution can appear.

Storyboard for	Darby Vale	Page 1 of 43
Client	William	Screen dimensions 1024 × 768



Purpose/description Introduction to Middle School page		Page title	Middle School
		File name	m_school
		Links	
		Link name	Link to
Graphics		Subjects	Table of subject
File name	Size (KB)	Year 7 & 8	Yr 7 & 8 page
DVC.jpg	23	VCE	VCE page
building.jpg	27	Policies	Policies page
heading.jpg	4	Contacts	Contacts page
		Site map	Site map page
Background(s)	Blue/white	Home	Home page

It also contains detailed information about:

- section name
- filename
- formatting features (e.g. background and foreground colours)
- purpose of the section
- media used in the section (e.g. images, videos)
- navigation to other sections of the solution.

An artistic endeavour at Keen College: Solution design

Now that the students have decided that a website is the best solution to the opportunity, they have to set about designing it so it will meet all the necessary functional requirements. They develop a site map to represent the structure of the website (Figure 6.15) and two layout diagrams to show how the final web pages will look (Figure 6.16 and Figure 6.17).

FIGURE 6.14
Storyboard for one page in a website

CASE STUDY



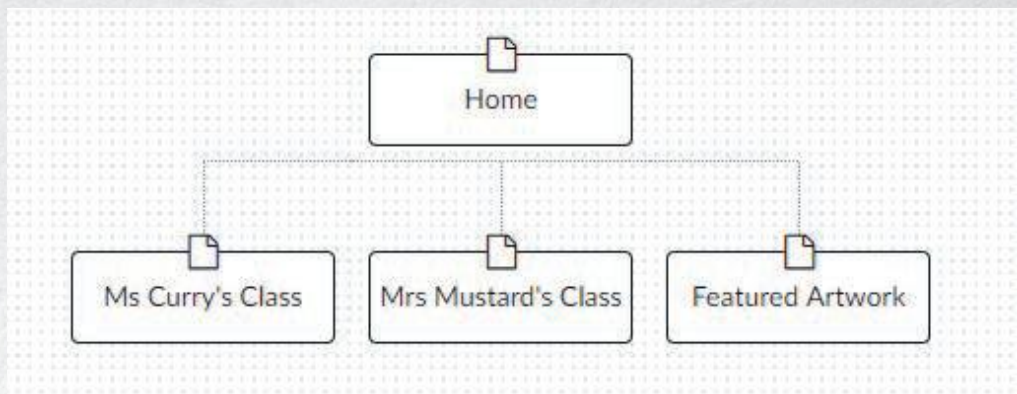


FIGURE 6.15 Site map of chosen web page solution

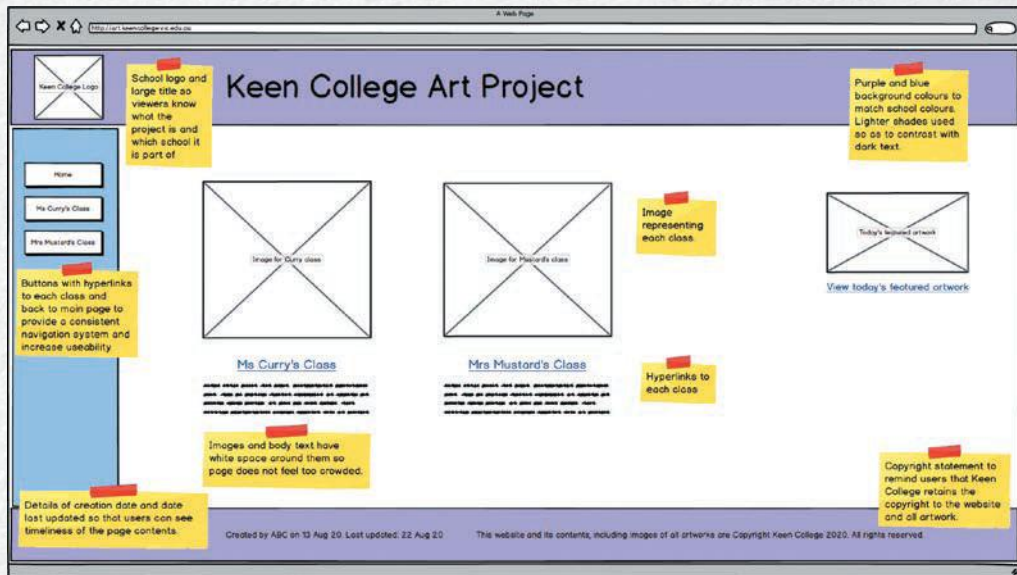


FIGURE 6.16 Layout diagram showing chosen web page solution

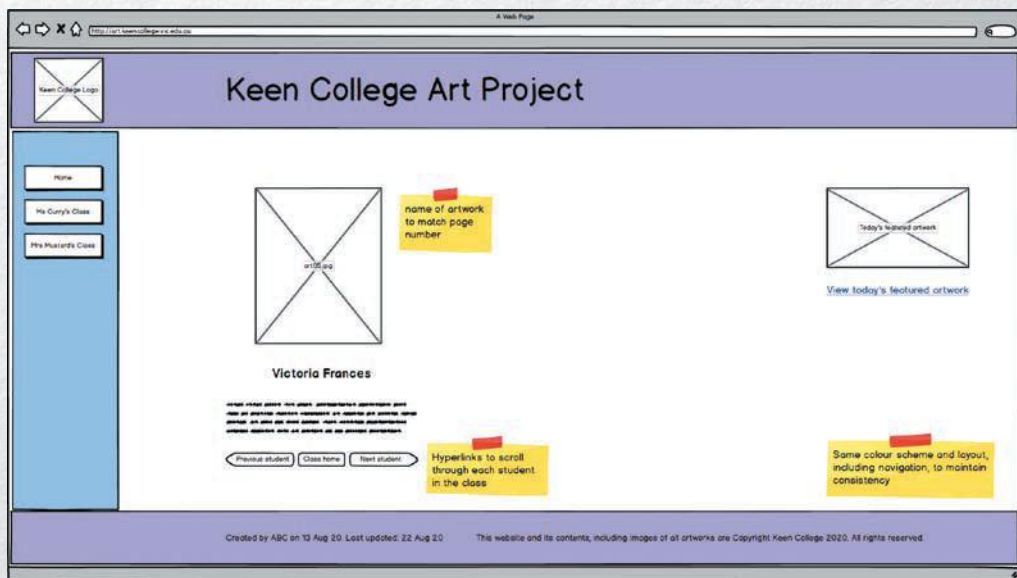


FIGURE 6.17 Layout diagram showing chosen web page solution

Methods for developing

Unit 2, Outcome 1, requires you to develop an innovative solution. Due to constraints, this may not involve creating the fully complete solution as envisaged, and may instead involve the creation of a proof of concept or prototype instead of a full product. Here is the difference between them.

Proof of concept

A **proof of concept** involves demonstrating whether an idea or concept is feasible, possible or viable. It is used to show, even in theory, whether a solution has the capability and the right characteristics to meet the need or opportunity identified.

Steps that may be involved in creating a proof of concept include:

- explaining the emerging technology being used
- discussing any issues (e.g. economic, legal or ethical) surrounding the technology
- identifying the need or opportunity
- identifying the solution requirements, constraints and scope of the solution
- generating solution designs and creating evaluation criteria
- identifying the inputs and output required
- identifying the processing required
- identifying the hardware and software required
- identifying any validation and testing needed
- explaining how the solution will achieve the evaluation criteria.

By following these steps, although there will not be a working solution, the intention is to provide enough evidence to show that the solution is possible in meeting the need or opportunity.

Prototype

A **prototype** is a model or a solution with limited functionality. The purpose of a prototype is to display the look and feel of the completed solution but without full functionality. It also allows a solution to be created, so users can give feedback and so changes can be made without wasting all of the time, effort and money that would be required to develop a completed solution (Figure 6.18).

A prototype differs from a proof of concept in that a prototype is built using the same software that will be used for the completed solution, whereas a proof of concept only provides evidence that a solution is feasible and may not involve the use of the solution software at all.

Product

A **product** is a completed solution with full functionality. All capabilities and features of the software identified during the analysis and design stages are included in the solution, all appropriate data is validated and all functions are fully tested.

A product differs from a prototype in that all the functions of the solution are working as expected, whereas in a prototype only a limited number of functions are present.

Many vehicle manufacturers create prototypes of new vehicles before they start full production to make sure that clients would be happy with the end product. If there are any client 'pain points' with the prototype, they can be addressed before the vehicle goes into mass production.

6.6 THINK ABOUT APPLIED COMPUTING

Find out two other industries that use prototypes and why they use them.

**CASE STUDY**

An artistic endeavour at Keen College: The product

The Keen College AC students have decided to create a functional website as their solution. This is a product that will incorporate all elements identified in the scope. While creating the product, students will prototype it to ensure that the basic structure of the template is sound and that the navigation is usable.

FIGURE 6.18

Prototyping a bionic exoskeleton in a robotics lab



Gorodenko / Shutterstock.com

Techniques for testing and evaluating innovative solutions

Testing and evaluating innovative solutions ensure that solutions not only meet the intended design objectives but are also effective, usable and valuable to the end user. Two fundamental techniques in this process are user testing and expert review.

User testing

User testing involves observing real users as they interact with the solution in a controlled environment or real-world setting. This technique is invaluable for gaining direct feedback on how well the solution meets user needs, its usability and overall user experience (UX). User testing can be conducted in various formats, depending on the objectives of the test. Two of these are prototyping and usability testing.

Prototyping

Prototyping involves creating simplified, often lower-fidelity versions of a product, enabling designers and developers to explore ideas and concepts without the need for complete development. Prototypes can range from paper sketches (low-fidelity) to interactive digital models (high-fidelity), depending on the stage of the development process and the specific goals of the testing.

Low-fidelity prototypes are quick and inexpensive to create, often consisting of sketches or paper models. They are useful for initial idea exploration and for communicating and refining basic design concepts and flow with stakeholders and team members.

High-fidelity prototypes are more detailed as well as being interactive, where the prototypes closely resemble the final product, providing a realistic experience of how the software or product will work. They are used for more advanced stages of testing and evaluation, particularly for assessing user interaction and engagement.

The benefits of prototyping include:

- the speed of idea exploration
- being able to provide visual/tangible representations of ideas for feedback
- being cost-efficient.

Usability testing

Usability testing is a user-centred technique that involves observing real users as they interact with the prototype to evaluate its usability, including ease of use, comprehension and ability to complete intended tasks. This testing can reveal insights into user behaviour, their preferences, and any potential areas of confusion. Detecting these early allows for refinement of the product before it is released to all potential users.

When conducting usability testing, the key is to focus on how easily users can navigate and use the solution to complete specific tasks. This can highlight usability issues that need to be addressed to improve the overall user experience.

The steps to take to conduct usability testing include: planning, recruitment, testing, analysis and iteration.

Planning

This is where objectives are defined and an appropriate method to test users is selected. The criteria for success of the usability tests is also determined in the planning stage. This includes deciding whether testing will be conducted in person or remotely, moderated or unmoderated.

Recruitment

The people who will be participating in the usability tests need to be determined and selected. Participants should represent the actual user base to ensure that the feedback is relevant. This involves recruiting a diverse group of users that matches the target demographics and user profiles.

Testing

The tasks that users complete when undertaking usability testing should be designed to reflect typical user activities. While they are using the software to undertake these activities, observers note where difficulties or confusion arise. The environment can vary from controlled, such as in a computer lab, to more natural settings, such as the user's home or workplace.

Analysis and iteration

The findings from usability testing are analysed to identify patterns and specific usability issues. These insights inform modifications to the prototype, leading to further rounds of usability testing and, as a result, the potential for further rounds of refinement.

The benefits of usability testing include:

- the level of user-centred feedback received
- increased efficiency in pinpointing specific areas where users might struggle
- increased chance of a higher-quality final product that has a high level of user friendliness.

Expert review

Expert review involves having usability experts or specialists in the relevant field review the solution against a set of established criteria. These experts rely on their knowledge and experience to identify issues and suggest improvements. The process can include heuristic evaluation, walk-throughs and compliance reviews.

In heuristic evaluation, experts evaluate the solution based on usability principles, also known as heuristics, such as those released by Nielsen. Table 6.6 outlines Nielsen's 10 usability heuristics that help to identify 'violations' within software as well as areas for improvement.

TABLE 6.6 Nielsen's 10 usability heuristics

Usability heuristic	Description
Visibility of system status	The system should always keep users informed about what is going on, through appropriate feedback within a reasonable time. This helps users understand the current context of their interaction with the system.
Match between the system and the real world	The system should use words, phrases and concepts familiar to the user, rather than technical terms. Follow real-world conventions, making information appear in a natural and logical order.
User control and freedom	Users often choose system functions by mistake and will need a clearly marked method of leaving the unwanted state without having to go through an extended process. Support undo and redo.
Consistency and standards	Users should not have to wonder whether different words, situations or actions mean the same thing. Follow platform conventions and industry standards.
Error prevention	Even better than good error messages is a careful design that prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
Recognition rather than recall	Minimise the user's memory load by making objects, actions and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
Flexibility and efficiency of use	Allow users to tailor frequent actions so that they can speed up an interaction (for example, keyboard shortcuts). Create a system that can cater to both inexperienced and experienced users.
Aesthetic and minimalist design	Information presented to the user should always be relevant and needed. Every extra unit of information presented competes with the relevant units of information and diminishes their relative visibility.
Help users recognise, diagnose and recover from errors	Error messages should be expressed in plain language (no codes), precisely indicate the problem and constructively suggest a solution.
Help and documentation	Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be taken and not be too large.

Read more about Nielsen's usability heuristics via the weblink.



Weblink
Nielsen's usability
heuristics

When conducting walk-throughs, specialists go through the solution, step by step, trying to perform tasks as a user would so that they can identify potential usability issues from a cognitive perspective.

Compliance reviews are important to ensure that relevant industry or domain requirements and laws are being adhered to. Assessing the solution against relevant standards and regulations ensures compliance with legal, security, accessibility and other requirements.

Testing

After designing and building your solution, you need to demonstrate that it has been thoroughly tested. You need to know what to test in your solution, so we plan the testing as part of the design stage.

If a solution fails, it could annoy or disadvantage users, so thorough and careful testing is necessary, whether the solution is a game, a website shopping cart, or an airliner's autopilot.

If your solution fails because of undiscovered faults, it may become difficult to use, or completely unreadable.

Testing checks that a solution does what it should do. Testing is not easy, quick or cheap – especially for a product such as an operating system, with megabytes of code in thousands of files created by hundreds of people.

The typical steps involved in testing are as follows.

- 1 Decide which tests will be conducted.
- 2 Create suitable test data.
- 3 Determine expected results.
- 4 Conduct the test.
- 5 Record the actual results.
- 6 Correct any errors.

There are many testing types, each intended to uncover different kinds of errors at different times during development. The types of testing relevant to your solution are listed in Table 6.7.

TABLE 6.7 Testing types

Type	What is tested?
Informal (alpha)	The part of the solution that has just been finished
User acceptance (beta)	Typical end users use their own equipment to check that the finished solution is acceptable under different user conditions
Component	A single part of a system works properly by itself (for example, a user entry form applies the correct delivery cost for a given destination)
Integration	Individual parts of a system work together (for example, the embedded Excel file correctly accesses the separate database table)
System	All components in the solution work properly as a single unit
Installation	The form control is installed correctly and working on your system, server or domain
Compatibility	The multimedia plug-in and its components are compatible with a variety of browsers and the main OS Note: ActiveX Form Controls will NOT work on the macOS
Usability	Whether users can operate your graphic solution quickly and simply
Accessibility	Whether users with special needs or disabilities can use your graphic solution

Test plan

A set of test data that will be used during development to ensure that the solution is functioning correctly is prepared in the design stage. The test data should be chosen to test all aspects of the solution, including identification and handling of unreasonable or incomplete data (**validation**). Once the solution has been shown to be functioning correctly, the test data is removed and the 'real' data relevant to the organisation is added. A test plan (developed during the design stage) is used to show all the functions to be tested, the sample test data and how the function is expected to handle the data. An example of a section from a test plan for a website is shown in Table 6.8. The actual testing process is covered in detail later in this chapter.



CASE STUDY

An artistic endeavour at Keen College: Test plan

TABLE 6.8 Part of the test plan for the Keen College website solution

What is to be tested?	Test data	Expected/required result	Why?
Link from home page to Ms Curry's subject page menu	Starting page = index.html Link = Ms Curry's class (curryclass.html) Destination page = curryclass.html	When the link 'Ms Curry's class' is selected, it should hyperlink to the page called 'curryclass.html'.	Because the link is to 'curryclass.html' and there is a page called 'curryclass.html'
Student artwork images load in the centre of the page	Page = curryclass05.html Image = art05.jpg (300 × 450 pixels) Alignment = 'center'	The image 'art05.jpg' must load in the centre of the page and aligned with the vertical middle of the text. Its height and width should be the full size of the image (300 × 450 pixels).	Because the code links to the 'art05.jpg' file in the images folder, its height is 450 and its width is 300 and its alignment is set to 'center'. This coding is placed before the text caption in the page body.
Screen layout on home page	Resolution = 1920 × 1080 keen_college_logo.jpg Navigation buttons (menu) Main text box	At 1920 × 1080 resolution, the Keen College logo and navigation buttons should appear on the left of the screen, separated from the main text by a border. The main text area should fit on the rest of the page without the need for scrolling.	Because the page elements were set to be viewed at 1920 × 1080 resolution
Page weight for index.html less than 200 KB	Reported file size of: index.html = 4 KB keen_college_logo.jpg = 40 KB Navigation buttons = 60 KB Main text area = 2 KB	Expected page weight = 4 + 40 + 60 + 2 = 106 KB, which is less than the 200 KB limit	Added up on a calculator
Home page download speed must be less than 2 seconds over NBN 25 Mbps connection	Page weight for index.html = 106 KB Theoretical NBN speed = 25 Mbps (8.3 MBps)	Expected download speed should be less than 2 seconds	Because $106 \div 8300 = 0.013$ seconds (on calculator), which is less than 2 seconds
User acceptance of functionality	Procedures: Navigate to the home page and locate the link to Ms Curry's class page. Navigate to the page for Ms Curry's class and locate the link to Victoria Frances's work.	Users should be able to perform each of the procedures listed with minimal difficulty.	Interfaces have been designed to allow for easy navigation to these key areas (navigation panel on left of main page and menu on class page).

Media and plug-ins

Each picture, audio clip, video, graph and animation (that is, any non-textual information) must be inspected to see that it is displaying in the right place, at the right time, and at the right speed and volume in a variety of common environments, meaning different browsers and devices.

Hyperlinks

Every internal and external link in the solution needs to be manually clicked and the result noted. Create a list of links and tick off each one as it passes testing.

Links to external services

You should be able to completely test all parts of the solution under your control. You need to test the operation of any external connections to your product, to ensure that data updates function as expected.

Readability

Use the checklist provided in Table 6.9 to test readability of your solution.

TABLE 6.9 Readability checklist

Checklist	Tick (✓)
Is the text large enough to read comfortably on a small device?	
Is contrast optimal, or at least satisfactory?	
Is the typeface a readable size?	
Are lines or paragraphs of a good length?	
Is text alignment attractive and readable on the page?	
Are the spelling, punctuation and grammar correct?	
Is the vocabulary appropriate and acceptable (inoffensive)?	
Is expression clear and unambiguous?	
Are headings clear and do they divide content into logical sections?	

Loading times

If the solution is online, clear your browser's cache to remove pre-loaded copies of files and media, and try loading the site via cable and Wi-Fi. Any page that takes more than a few seconds to load should be inspected and optimised. Another method is to use one of many online services that can measure the loading times for your pages. Online data repositories may have varying access times due to user demand.

Browser compatibility

Does your solution rely on a browser for presentation? Check that plug-ins and installed players and codecs (coder/decoders or compressor/decompressors) can read and display your chosen media. Browsers differ in their ability to interpret different media, and some systems may not have the right technology, such as HTML5, or the necessary plug-ins installed, such as Silverlight. Every piece of media must be checked on the dominant browsers to verify that they appear as expected. Remember that Flash will not play at all on many mobile devices.

You can manually test most site functionality yourself. If your solution is online, however, there are many services that can perform automated cross-browser compatibility checks using many new and previous browser versions.



Weblink
W3C

Accessibility

Does your solution create unnecessary difficulty for users with poor eyesight or muscular control, or weak language skills, or other common disabilities? Is **alt text** applied to images? Are colour combinations considerate of colourblind people? Many colourblind-safe palettes are documented online.

There are several places online to test the accessibility of your solution. Try the World Wide Web Consortium (W3C) website.

Dynamic features

Every selection option item must be checked and its behaviour documented in a **testing table** (see Table 6.10). If data entry forms are expected to work, data should be entered and its successful arrival at its destination should be documented. Any simulated functionality, such as a faked login box, should, as far as is practical, appear to work genuinely. Any coding, such as JavaScript, PHP/MySQL, Java, PERL, macros and Python, should be run using a variety of test data and the behaviour of the code recorded.

Classroom constraints

Your dynamic graphic solution may, because of constraints in classrooms and networks, not have access to updated data online. However, it should have the look, feel and apparent functionality of a real online solution, even though some features may have to be simulated because it is unreasonable to expect them to function under all working conditions.

Testing the solution

While a solution is under development, it will undergo extensive testing to ensure that it functions as expected. Much of the testing may be informal, but for the client a set of formal testing documentation needs to be created. You may make use of a testing table to summarise certain tests, and annotated screenshots to demonstrate the results of other tests.

Testing table

A testing table is a commonly used way to record evidence of functionality testing. A testing table for a smart watch might look like the one shown in Table 6.10.

TABLE 6.10 A testing table

What was tested	How it was tested	Expected result	Actual result	How it was fixed, if relevant
Ability to access apps on the watch by spinning a dial	Spun the dial on the watch to see if it cycled through apps	Apps will cycle one by one	Apps cycled one by one	N/A
Pressing on an app icon on the watch face will open the app	Open each app by pressing on the app icon	Each icon will open the relevant app	Apps opened but some apps crashed	Not yet fixed
Readability	Asked five volunteers to read the text on the watch face and report on text size, contrast, alignment	Reports that text was easy to read	One volunteer suggested that italic fonts were hard to read	Removed all italic fonts from the watch system
Loading times for apps	Performed a speed test on loading of each app by recording the load time	All apps passing the speed test	All apps passed the speed test	N/A
Watch tracking number of steps taken by wearer	Asked a volunteer to wear the watch and take 100 steps	Watch should record 100 steps	Watch recorded 100 steps	N/A

How to document your testing

- Use a testing table (such as the one shown in Table 6.10).
- Seek a subjective report from a fellow student who tried out your solution's readability and usability.
- Capture screenshots of features that are not normally visible, such as drop-down menus and warning messages, showing that they work when needed.
- Make handwritten calculations, annotating printouts of screenshots of your solution's calculations to verify that the output has been checked for accuracy.
- Capture screenshots of the solution's validation rules responding properly to invalid data.

An artistic endeavour at Keen College: Validating and testing the solution

The content for the website is validated by the AC students. They check that there is a blurb and an image for every student. Then they manually proofread the text, looking to see that it makes sense and to correct any obvious spelling errors. Students also use an electronic spelling and grammar checker to assist with this process. If they are unsure of the spelling of a name, or of the description of an artwork, they ask the relevant student in order to clarify what the text is meant to be.

Once the website structure is complete and the images and text are placed in relevant folders, several students begin the task of formally testing the product. Using the test plan created in the design stage, the students check for such things as links working properly and that the correct images are loading on each page. They make use of both a testing table and annotated screenshots to test the website thoroughly.

TABLE 6.11 Keen College testing table

What is to be tested?	How it was tested/test data	Expected/required result	Actual result	How it was fixed, if relevant
Link from homepage to Ms Curry's subject page menu	Starting page = index.html Link = Ms Curry's class (curryclass.html) Destination page = curryclass.html	When the link 'Ms Curry's class' is selected, it should hyperlink to the page called 'curryclass.html'	The link was broken and went to curryclas.html	Changed the link to be curryclass.html
Student artwork images load in the centre of the page	Page = curryclass05.html Image = art05.jpg (300 × 450 pixels) Alignment = 'center'	The image 'art05.jpg' must load in the centre of the page and aligned with the vertical middle of the text. Its height and width should be the full size of the image (300 × 450 pixels)	The image loaded in the centre of the page and was aligned with the vertical middle of the text. The height and width were correct.	N/A

CASE STUDY



**TABLE 6.11** Keen College testing table (*continued*)

What is to be tested?	How it was tested/test data	Expected/required result	Actual result	How it was fixed, if relevant
Screen layout on home page	Resolution = 1920 × 1080 keen_college_logo.jpg Navigation buttons (menu) Main text box	At 1920 × 1080 resolution the Keen College logo and navigation buttons should appear on the left of the screen, separated from the main text by a border. The main text area should fit on the rest of the page without the need for scrolling.	The logo and navigation appeared on the left, separated by a border. The main text area did not fit on the rest of the page – it needed scrolling.	Reformatted the main text so that scrolling no longer occurred
Page weight for index.html less than 200 KB	Reported file size of: index.html = 4 KB keen_college_logo.jpg = 40 KB Navigation buttons = 60 KB Main text area = 2 KB	Expected page weight = 4 + 40 + 60 + 2 = 106 KB, which is less than the 200 KB limit	Actual weight was 106 KB, less than the 200 KB limit	N/A
Home page download speed must be less than 2 seconds over NBN 25 Mbps connection	Page weight for index.html = 106 KB Theoretical NBN speed = 25 Mbps (8.3 MBps)	Expected download speed should be less than 2 seconds	Download speed was almost instant, much less than 2 seconds	N/A
User acceptance of functionality	Procedures: Navigate to the Home page and locate the link to Ms Curry's class page. Navigate to the page for Ms Curry's class and locate the link to Victoria Frances's work.	Users should be able to perform each of the procedures listed with minimal difficulty	Navigation to Victoria Frances's work was quick and easy	N/A

Evaluation criteria

While **evaluation** is the final stage of the problem-solving methodology, the **evaluation criteria** are developed during the design stage. The purpose of evaluation is to check how well the solution is satisfying the needs of the user for which it was originally created.

Evaluation is not the same as testing; its purpose is distinctly different. By the time evaluation begins, the solution has already been proved to work properly and its functionality is no longer in question.

Evaluation can best be understood by saying what it does not do.

- Evaluation does *not* test that a solution is working properly. That should have been done during testing.
- Evaluation does *not* enter test data to check that output is accurate. That should have been done during testing.
- Evaluation does *not* use a stopwatch to time how long a process takes. That should have been done during testing.
- Evaluation does *not* perform checks with immediate results, such as pulling out the power plug to see if a system loses data. That should have been done during testing.

Evaluation looks at a solution's performance *over time* in terms of the evaluation criteria.

What to evaluate

Evaluation criteria are determined during the design phase of the problem-solving methodology and are based on the most important qualities that the solution is expected to have when it is designed.

Evaluation criteria fall under two headings: efficiency and effectiveness.

- 1 **Efficiency** can be measured in terms of speed or productivity (work produced in a given time), profitability (income generated versus running costs) and labour requirements (how much labour is required to achieve its productivity levels).
- 2 **Effectiveness** includes completeness, readability, attractiveness, clarity, accuracy, accessibility, timeliness, communication of message, relevance and usability.

6.7 THINK ABOUT APPLIED COMPUTING

How do you evaluate a new website, game or social media app? On what measures do you base your opinion?

An artistic endeavour at Keen College: Evaluation criteria

The Keen College AC students have created the following evaluation criteria for the planned website solution. They plan to perform the evaluation during the school holidays so they can respond to feedback when they get back.

TABLE 6.12 Evaluation criteria

Criterion	Time frame	Data required	Data-collection method	Assessment	Judgement
All student artwork can be viewed	During school holidays	Web page hit counter	Website log files	Check website logs to see if the artwork of each student has been accessed at least once.	If the logs show that all student artwork has been accessed at least once, then the criterion is achieved.

CASE STUDY



**TABLE 6.12** Evaluation criteria (continued)

Criterion	Time frame	Data required	Data-collection method	Assessment	Judgement
Artwork images are downloaded within an acceptable time frame	During school holidays	Teacher, student and parent opinions	Online survey	Ask teachers, parents and students if the downloading of artwork images occurs within an acceptable time frame.	If 95% of feedback states that the images are being received within an acceptable time frame, then the criterion is being achieved.
Website is easy to use to find student artwork	During school holidays	Teacher, student and parent opinions	Online survey	Ask teachers, parents and students to rate the usability of finding student artwork out of 10.	If the average rating is 8 or better, then the criterion is being achieved.

Evaluating the solution

During the design stage, you create the evaluation criteria. This helps to guide the design and development of the solution. Once the solution has been finalised and installed, the formal evaluation process can be conducted. The evaluation might take place 3–6 months after the solution is implemented so that people have time to learn the new system and to give time for any minor bugs to be dealt with.

Evaluation methods

For each evaluation criterion, there must be a corresponding evaluation method that can measure the degree to which the criterion has been achieved.

- **Objective** (fact-based, measurable) results are solid facts that are hard to argue with. Measure whenever you can.
- **Subjective** results (emotions, opinions, personal judgements) can be gained from interviews, questionnaires and surveys. These should only be used when objective measurement is not possible or practical.

TABLE 6.13 Typical evaluation methods

Criterion	Method
Accuracy (effectiveness)	Check the complaints log and count the complaints from staff or customers about inaccurate information received over three months from the system.
Reliability (effectiveness)	Count the number of faults in the system's error log.
Security (effectiveness)	Count the number of successful and thwarted attempts made to penetrate system security.

TABLE 6.13 Typical evaluation methods (continued)

Criterion	Method
Attractiveness, pleasure, comfort, confidence (effectiveness)	Interview users.
Productivity (efficiency)	Refer to system logs and count how many transactions the system handled over three months compared with the previous system.
Profitability (efficiency)	Ask the accountants to tally the new system's running costs over time. Check organisational profit figures and see if profit has increased.
Labour requirements (efficiency)	Count the number of staff hours spent operating and maintaining the system compared with the previous system.
Ease of use, usability (effectiveness)	Count the number of times online help was used (indicating that the solution may not have been intuitive). Add up how many errors were made by users. (A solution that is hard to use tends to cause users to make mistakes.) Check the help desk records to see how often users asked for help or complained about the solution. Give users a questionnaire asking about their feelings regarding the system's usability.

Remember: evaluation assesses your solution's performance over time. It is not instantaneous in the same way that testing is. Any emotional or judgemental feedback is only gathered on appropriate criteria. For example, it is pointless to ask interviewees questions such as, 'Is the new system faster than the old one?' Even if you received an answer to this question, you would not be able to trust its accuracy.

When to evaluate

Evaluation occurs after the solution has been in regular use for some time, so the solution is well 'bedded in' and its users are familiar and comfortable with it. A few months of regular, daily use is typical.

Evaluating a solution too soon can lead to negative feedback if users are not yet used to it and so are slow and prone to making errors. Later, when they are comfortable and skilled with the solution, their feedback may be much more positive.

In cases when a system is used infrequently, but its success is critical to the organisation (such as creating school reports), evaluation may be done immediately after the system is used.

An artistic endeavour at Keen College: Evaluation

Once the website solution has been completed and put onto the Keen College web server, it is made available to parents, students and teachers. The students allow all users time to learn to use and navigate the website during the school holidays, before conducting an evaluation. In their evaluation strategy, the students have decided to survey all user groups to get feedback on the website's usability. The questions are based around the evaluation criteria created during the design stage. As a result of the evaluation process, the students identify a few small changes to make to the website, but overall it appears to be meeting the requirements of the users.

6.8 THINK ABOUT APPLIED COMPUTING

Why do software developers wait a period of time after the product is released before completing an evaluation?

CASE STUDY



**TABLE 6.14** Keen College evaluation results

Criterion	Time frame	Data required	Data-collection method	Assessment	Judgement
All student artwork can be viewed	During school holidays	Web page hit counter	Website log files	Check website logs to see if the artwork of each student has been accessed at least once.	The logs showed that every student page had been accessed at least once. This criterion was achieved.
Artwork images are downloaded within an acceptable time frame	During school holidays	Teacher, student and parent opinions	Online survey	Ask teachers, parents and students if downloading artwork images occurs within an acceptable time frame.	96% of feedback indicated that downloading artwork was performed in an acceptable time frame. This criterion was achieved.
Website is easy to use to find student artwork	During school holidays	Teacher, student and parent opinions	Online survey	Ask teachers, parents and students to rate the usability of finding student artwork out of 10.	The average rating for ease of use was 9. This criterion was achieved.

6

CHAPTER SUMMARY

Essential terms

Act of Parliament a document that contains a set of conditions and rules about a particular area or topic

adjustments changes made to the project plan as a result of events that may have occurred as the project progressed

alt text text descriptions added to images on a web page, which describe the image for users who may not be able to access the image; 'alt' is an abbreviation of 'alternative'

annotations comments or notes placed on a Gantt chart to explain any changes that were made

biometric systems security and authentication systems that use physiological and behavioural characteristics, such as fingerprints, facial recognition or voice patterns, to verify the identity of individuals

brainstorming a technique to generate ideas to solve a problem

copyright gives the owner of an original work (e.g. software application, movie, song or images) the right to control who can use the original work, and how

Copyright Act 1968 an Australian federal law containing conditions and rules related to copyright

dependencies relationships between tasks, indicating when a task should begin or end in relation to another task or group of tasks

effectiveness how well a solution meets the needs of its users, measured in terms of completeness, readability, attractiveness, clarity, functionality, accuracy, accessibility, timeliness, report formats, relevance, usability and communication of message

efficiency a measure of how much time, cost and effort are required to achieve the intended results

ethical issue a situation that arises when current behaviours or practices could be considered to be the wrong thing to do

ethics accepted moral standards that guide behaviour; these standards may be common across a particular society or specific to a single organisation, and they apply to questionable activities over and above any legal requirements; ethics often provide us with a set of guidelines for appropriate behaviour

evaluation the use of criteria created during the design stage to evaluate or judge whether the solution is meeting the needs of the user; the final stage of the problem-solving process

evaluation criteria measures that are used to assess or judge whether a solution is meeting the user's needs and/or requirements

federal parliament a body of elected representatives responsible for creating legislation that applies to all individuals and organisations within Australia

functional requirements requirements of a digital solution that are directly related to what the solution will do

Gantt chart a project management tool that displays project schedule information by listing project tasks and their corresponding start and finish dates in a calendar format

Health Records Act 2001 a Victorian state law containing conditions and rules related to how and when an individual's personal health data and information can be collected by both Victorian government agencies and private organisations that either offer health services or handle health records

innovative solution a unique, original and previously unknown way to solve a problem

interoperability the ability of a device to connect and interact with other devices, particularly devices belonging to different information systems

interview a conversation between two or more people where one person (the interviewer) asks questions and the others (the interviewees) give their answers to the question

job displacement involuntary job loss due to technological, organisational or economic changes

legislation a body of laws, created by governments, that individuals and organisations resident in the government's jurisdiction must follow

local councils elected bodies that can create by-laws, which apply within each individual local council area

mind mapping a technique for quickly generating and linking ideas

need something that it is essential to the solution so that the user may complete the task

non-functional requirements characteristics that the user or client would like the solution to have; often tied to solution constraints

objective not influenced by personal feelings or prejudice; unbiased

opportunity something that occurs when circumstances or factors are present that may allow for a new solution, which does not already exist, or which people have not yet thought of

Privacy Act 1988 an Australian law that regulates the handling of personal information about individuals

Privacy and Data Protection Act 2014 a Victorian law that regulates the use, protection and storage of data and information in Victoria

product a completed solution with full functionality

project a series of interrelated tasks that need to be completed to be able to create a solution to a need or opportunity

project journal similar to a diary in which events that affect each task in a project are written down to create a history of the project

proof of concept the process of demonstrating whether an idea or concept is feasible, possible or viable

prototype a model or solution with limited functionality

pseudocode a design tool combining the structure of programming language code with instructions written in plain English, used to represent the logic or steps of processing that occur when a solution is run or executed

scope of the solution boundaries or parameters outlining what the solution will contain

site map a way of displaying how information has been structured within a solution and how a user can navigate to find the required information

solution a way to solve a problem

solution constraints factors that may limit or restrict the solution requirements, both functional and non-functional

solution requirements what the client needs from the solution

state parliaments bodies of elected representatives responsible for creating legislation that applies to all individuals and organisations within each particular state

storyboard a tool used to show the features of an individual section of a solution

subjective based on feelings and emotions, opinions and personal judgements

survey a series of predetermined questions that can be sent to a participant to answer

task delegation assigning responsibilities among team members

testing the process of checking that the innovative solution is working as expected

testing table a commonly used way to record evidence of functionality testing

validation the process of inspecting data that is being input into a solution to check if the data is reasonable

version control a system that manages changes to documents, computer programs and other information, tracking revisions and enabling collaboration

Important facts

- 1 Legislation is a body of laws, created by governments, that individuals and organisations resident in the government's jurisdiction must follow.
- 2 Federal parliament is responsible for creating legislation that applies to all individuals and organisations within Australia. State parliaments are responsible for creating legislation that applies to all individuals and organisations within each particular state, such as New South Wales and Victoria, while local councils create by-laws that apply within each individual local council area.
- 3 The *Privacy Act 1988* is an Australian federal law. It contains conditions and rules related to how and when personal data and information can be collected by Australian government agencies and private organisations that turn over more than \$3 million annually.
- 4 The *Privacy and Data Protection Act 2014* is a Victorian state law. It contains conditions and rules related to how and when personal data and information can be collected by Victorian government agencies and private organisations that conduct contract work on behalf of the Victorian Government.
- 5 The *Health Records Act 2001* is a Victorian state law containing conditions and rules related to how and when an individual's personal health data and information can be collected by both Victorian government agencies and private organisations that either offer health services or handle health records.
- 6 The *Copyright Act 1968* is an Australian federal law containing conditions and rules related to copyright. Copyright gives the owner of an original work (e.g. software application, movie, song or images) the right to control who can use the original work, and how it can be used.
- 7 Ethics are beliefs, principles or standards that individuals, organisations and society have regarding acceptable behaviour.
- 8 Techniques used to collect data include observation, interviews, questionnaires and surveys.
- 9 A solution involves formulating a way to solve a problem. A need is something that is essential for a user to complete a task. An opportunity occurs when circumstances or factors are present that may allow for a new solution, that does not already exist, or that people have not even thought of yet.
- 10 To qualify as an innovative solution, a solution must be unique and developed independently.
- 11 There are a variety of techniques available to generate solution ideas, such as brainstorming and mind mapping.
- 12 A successful problem-solver will consider current functional and non-functional requirements and relevant constraints to develop an imaginative range of options from which the best solution can be chosen and developed into a detailed design.
- 13 Types of design tools include (but are not limited to):
 - layout and mock-up diagrams (see Chapter 2)
 - pseudocode
 - site maps
 - storyboards.
- 14 Validation checks the reasonableness of data inputs.
- 15 Testing checks the accuracy of information outputs.
- 16 A proof of concept involves demonstrating whether an idea or concept is feasible, possible or viable.
- 17 A prototype is a model or a solution with limited functionality.
- 18 A product is a completed solution with full functionality.
- 19 Evaluation is the final stage of the problem-solving methodology. It checks how well the solution is satisfying the needs of the user for which it was originally created. Evaluation occurs after the solution has been implemented.

- 20** Efficiency can be measured in terms of speed or productivity (work produced in a given time), profitability (income generated versus running costs) and labour requirements (how much labour is required to achieve its productivity levels).
- 21** Effectiveness includes completeness, readability, attractiveness, clarity, accuracy, accessibility, timeliness, communication of message, relevance and usability.
- 22** A Gantt chart provides a standard format for displaying project schedule information by listing project tasks and their corresponding start and finish dates in a calendar format.
- 23** A project journal can be used to document the development of the solution.



Key legislation and emerging technologies

- 1 Differentiate between federal, state and local laws.
- 2 Explain the relationship between an Act of Parliament and law.
- 3 List four Acts of Parliament that have an effect on emerging technologies.
- 4 Explain how each Act of Parliament you listed affects two chosen emerging technologies.
- 5 What is meant by an ethical issue?
- 6 Discuss one ethical issue related to the Internet of Things (IoT).

Collecting data

- 7 Explain why it is important to use a number of data-collection techniques when determining a user's needs.
- 8 What is the analysis stage of the problem-solving methodology about?
- 9 Explain the type of data that can be collected via both interview and survey.

Solution specifications

- 10 Explain how functional requirements differ from non-functional requirements.
- 11 Explain how two different types of constraints may affect solution requirements.
- 12 Discuss why it is important to define the scope of a solution.

Characteristics of solutions

- 13 Define an 'innovative solution'.
- 14 Explain how a need differs from an opportunity.
- 15 List the four steps in devising an innovative solution.
- 16 List three techniques used to generate creative ideas. Give an example of one of these techniques.
- 17 Explain a technique to select the best solution from a list of three.

Representing solution designs

- 18 List the two activities involved in the design stage of the problem-solving methodology.
- 19 Explain the purpose of pseudocode.
- 20 What is the difference between a site map and a storyboard?
- 21 Explain how a storyboard can be used to represent both the appearance and the functionality of a solution.



TEST YOUR KNOWLEDGE

Validating and testing solutions

- 22** Explain how validation differs from testing.
- 23** List the four column headings of a testing table.

Developing your solution

- 24** Describe what occurs in the development stage of the problem-solving methodology.

Evaluating the solution

- 25** Describe the purpose of evaluation criteria.
- 26** List the tasks involved in an evaluation strategy.

Coordinating and monitoring projects

- 27** List four elements of a project.
- 28** Describe the features found on a Gantt chart.
- 29** Explain the purpose of a project journal.
- 30** Describe an advantage of annotations and adjustments of project plans.



A healthcare facility wants to implement a new system to improve the quality of care for elderly patients. It would like to use some type of robotic system with sensors and AI to monitor patients' health, provide medication reminders, assist with daily activities and alert staff to any emergencies. The goal is to enhance patient safety, improve the efficiency of healthcare delivery and provide personalised care. Your team has been tasked with proposing an innovative solution for this robotic health assistant system, considering the diverse needs of elderly patients, healthcare staff and the facility's operational requirements.

- 1 Explain the techniques you would use to collect data.
- 2 Use the problem-solving methodology to analyse the healthcare facility's needs and requirements.
- 3 Identify the opportunity.
- 4 Specify the functional and non-functional requirements.
- 5 What are the constraints?
- 6 Use the solution specifications and constraints to determine the scope of the project.
- 7 Present the healthcare facility with three possible innovative solutions.



PREPARING FOR Unit

2

OUTCOME 1

Students are to work collaboratively to analyse, design, develop and evaluate an innovative solution to an identified need or opportunity involving a digital system.

Steps to follow

- 1 Create groups within the class.
- 2 Discuss a range of needs and opportunities and how each could be solved using emerging technology.
- 3 Select a topic.
- 4 Complete an analysis by identifying the solution specifications: functional and non-functional requirements, constraints and scope.
- 5 Complete the design of the solution by creating design tools that show the appearance and/or functionality of the proposed solution.
- 6 Complete the development of the solution by creating the solution using technology, including validation and testing.
- 7 Complete the evaluation of the solution by creating evaluation criteria and then collecting data that can be used to measure whether the criteria were achieved.

Documents required for assessment

Your teacher will advise you how to submit each section of the project. Elements that will require submission are:

- project plan
- solution specifications
- design tools
- evidence of the completed solution
- evaluation report.

KEY KNOWLEDGE

On completion of this chapter, you will be able to demonstrate knowledge of:

- KK2.2.1: emerging trends in cyber security
- KK2.2.2: functions and characteristics of key hardware and software components of networks required for communicating and storing data and information
- KK2.2.3: strengths and limitations of wired, wireless and mobile communications technology, measured in terms of cost, data storage options, reliability and security
- KK2.2.4: technical underpinnings of intranets, the internet and virtual private networks
- KK2.2.5: applications and capabilities of Local Area Networks (LANs), Wide Area Networks (WANs) and Wireless Personal Area Networks (WPANs)
- KK2.2.6: risks of using networks in a global environment
- KK2.2.7: technical underpinnings of malware that can intentionally threaten the security of networks
- KK2.2.8: security threats to data and information on networks
- KK2.2.9: practices for reducing risks and mitigating threats to networks
- KK2.2.10: cryptographic techniques to protect data and networks
- KK2.2.11: emergence of artificial intelligence in providing network security mechanisms
- KK2.2.12: the role of ethical hacking, such as identifying vulnerabilities and weaknesses in networks
- KK2.2.13: key legislation and industry frameworks that affect how organisations ethically control the security and communication of data and information.

Victorian Curriculum and Assessment Authority, Victorian Certificate of Education Applied Computing Study Design 2025–2028, VCAA, 2024, pp. 34–5

FOR THE STUDENT

This chapter provides an overview of networks, with an emphasis on network security in a global environment. An explanation of networks, key network hardware and software components and communication methods is given. A range of security threats to data and information are discussed, in addition to a range of security measures that can be used to protect data and information from these security threats. The interactions and impacts created by networks are also discussed, with a focus on legal and ethical considerations required by organisations that operate in a global environment. For this outcome, you will be required to respond to a teacher-provided case study, analysing a network for vulnerabilities or particular security incident. As part of this, you will evaluate the threats to the network and provide suggested strategies to protect data and information that uses that network.



For Unit 2, Outcome 2, you are required to describe the capabilities of different types of networks.

THINK ABOUT APPLIED COMPUTING

7.1

List two advantages and two disadvantages of a standalone device.

Key network hardware and software components

A **standalone device** is a self-contained device or system in that it can perform its function without the need of another device, computer or connection. The device does not communicate with any other device.



FabrikaSimf/Shutterstock.com

FIGURE 7.1 A standalone fax machine; it does not need to be connected to a computer or network to perform its function.

THINK ABOUT APPLIED COMPUTING

7.2

To check the download speed of your home internet connection, visit the speed test weblink and run the speed test.



Weblink
Speed test

A **network** is a collection of computers and devices connected by communications channels that facilitates communications among users and allows users to share resources with one another. Examples of such resources are data, information, hardware and software. Networks can be internal to an organisation, or may cover the whole world by connecting to the internet.

Networks exist for sharing information, such as spreadsheet files, database records, email – indeed, anything that helps someone get their job done. The ability to share resources, such as servers, printers and software, also makes a network valuable.

As shown in Figure 7.2, for successful communications, a network needs:

- a sending device, such as a notebook computer, which initiates an instruction to transmit data, instructions or information
- a communications device, such as a wireless adaptor inside a notebook computer, to forward packets of data, instructions or information from a sending device via signals carried by a communications channel
- a communications channel or transmission media, such as a cable or radio waves, through which the digital signals travel
- a communications device, such as a wireless router, that receives the signals from the communications channel and forwards the packets to the receiving device
- a receiving device, such as a printer, which accepts the data, instructions or information.

Sending devices can usually also serve as receiving devices. Examples include notebook computers, personal computers, tablets and mobile phones. Such devices usually have a built-in communications device.

The primary function of a communications device such as a broadband router is to transmit data, instructions and information between a sending device and a receiving device. Data, instructions and information travel along a communications channel in the form of a digital signal.

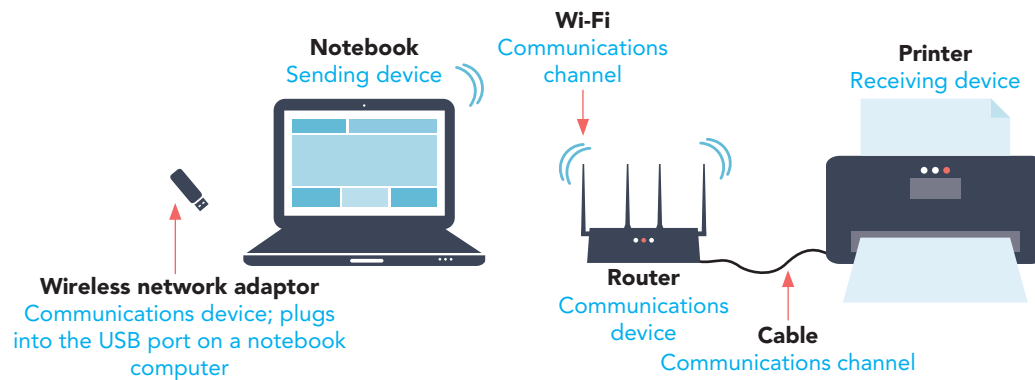


FIGURE 7.2 The notebook sends an instruction to the wireless adaptor (communications device), which sends a signal over radio waves (communications channel). The router (communications device) receives the signal and sends an instruction to print via cable (communications channel) to the printer (receiving device).

A digital signal consists of individual electrical pulses that represent the bits grouped together into bytes. Early networks used analog signals that consisted of a continuous electrical wave. Computers process data as digital signals, so a device was needed to convert between analog and digital signals. That device is a **modem**.

Advantages of networks

Advantages of networks include their ability to:

- share data and information – once devices are connected, they are able to share data and information. Files on one device may be accessed by other devices on a network
- allow communication – users on network devices can communicate with each other. This may involve email, video conferencing, online chat or accessing shared calendars
- share hardware and software – once devices are connected and can communicate, they are also able to share other digital system resources.

Types of networks

Networks can often be grouped into two types. The type of network refers to the area over which the network provides connectivity. A network is either a Local Area Network (LAN) or a Wide Area Network (WAN).

Local Area Network

A **Local Area Network (LAN)** connects computers and devices in a limited geographical area, such as a home, school, office building (Figure 7.3) or closely positioned group of buildings. Each computer or device on the network is a **node**. In many networks, nodes are connected to the LAN via cables; many new networks, however, use wireless transmission media. A wireless LAN (WLAN) uses no physical wires; instead, it uses wireless media, such as radio waves. Computers and devices that access a WLAN must have wireless capability, whether that is built-in or ad hoc using a USB wireless receiver. Usually a WLAN communicates with a wired LAN for access to its resources, such as software, hardware and the internet.

7.3 THINK ABOUT APPLIED COMPUTING

List two benefits of connecting two or more devices together.

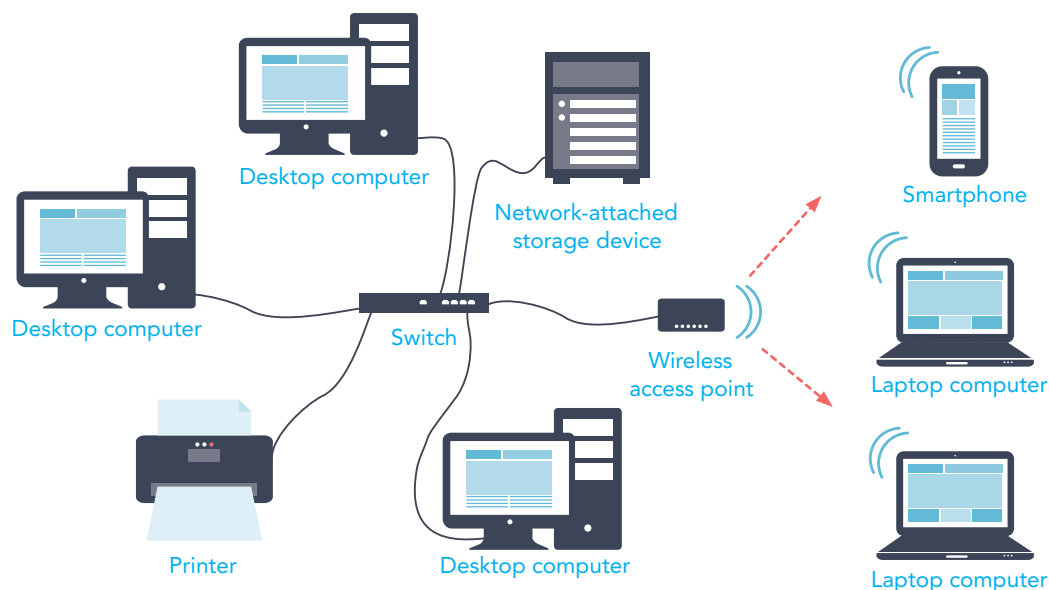


FIGURE 7.3 A Local Area Network operating within a confined geographical area



WebLink
Network World

The logical design of the components of the network, including the number and types of servers, workstations and network resources, is known as the network architecture. It includes the communications devices and the types of physical and wireless transmission media used to connect components. Reviews of network operating systems and general information on networks can be found on the Network World website.

Home networks

If you have multiple computers in your home or home office, you can connect all of them together with a home network (see Figure 7.4). The advantages of a home network include the following:

- Desktop computers, notebooks, tablets and smartphones can all communicate with each other.
- All the computers can be connected to the internet at the same time.
- All computers can share peripherals, such as a scanner, printer or a network-attached storage device.
- Each networked computer is able to play multiplayer games with players on other computers in the house.
- Smart TVs can connect to the internet.

Wide Area Network (WAN)

A **Wide Area Network (WAN)** is situated in more than one geographical location. Often, it consists of two Local Area Networks in different locations that are able to communicate with each other using an internet connection. For example, two offices, each with its own Local Area Network, can communicate with each other using an internet connection (Figure 7.5).

One difference between a Local Area Network and a Wide Area Network is that a WAN generally uses third-party transmission media, often an internet connection, to connect the two locations. Third-party transmission media are transmission media that belong to another organisation. Often, this will be a telecommunications organisation such as Telstra or Optus. Once an organisation uses third-party transmission, it starts to lose some control over the network and the potential for data threats increases.

THINK ABOUT APPLIED COMPUTING

7.4

Describe why an organisation that has offices in two different cities would use third-party transmission media rather than install its own dedicated connection.

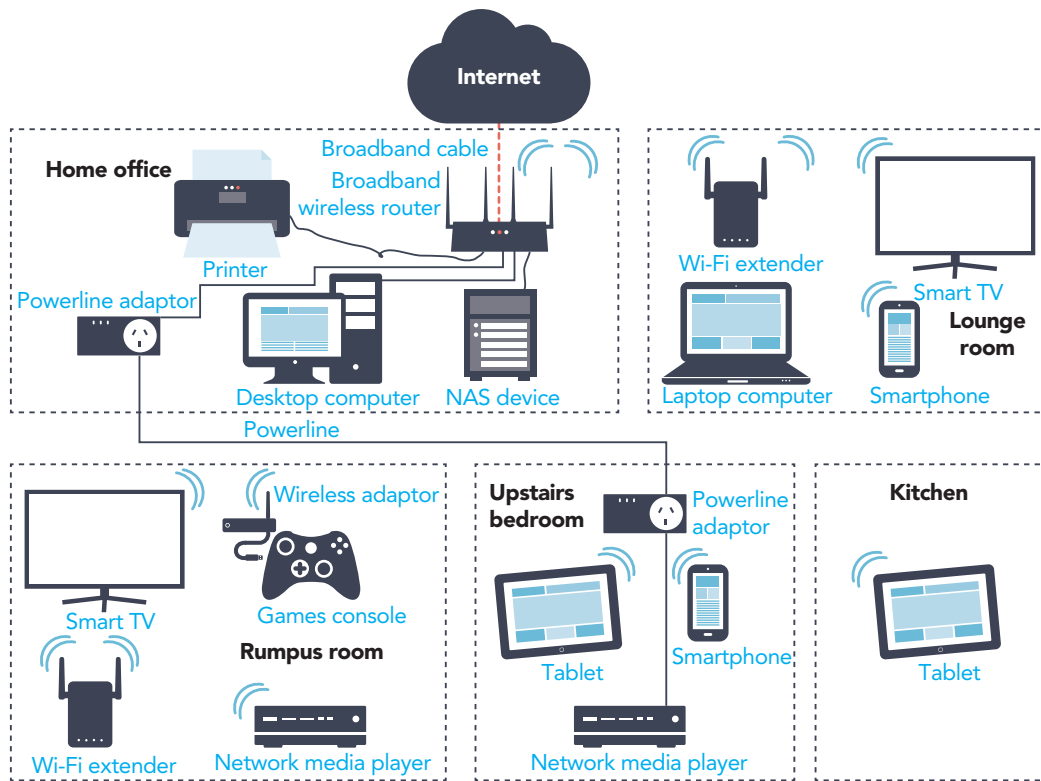


FIGURE 7.4 A home network

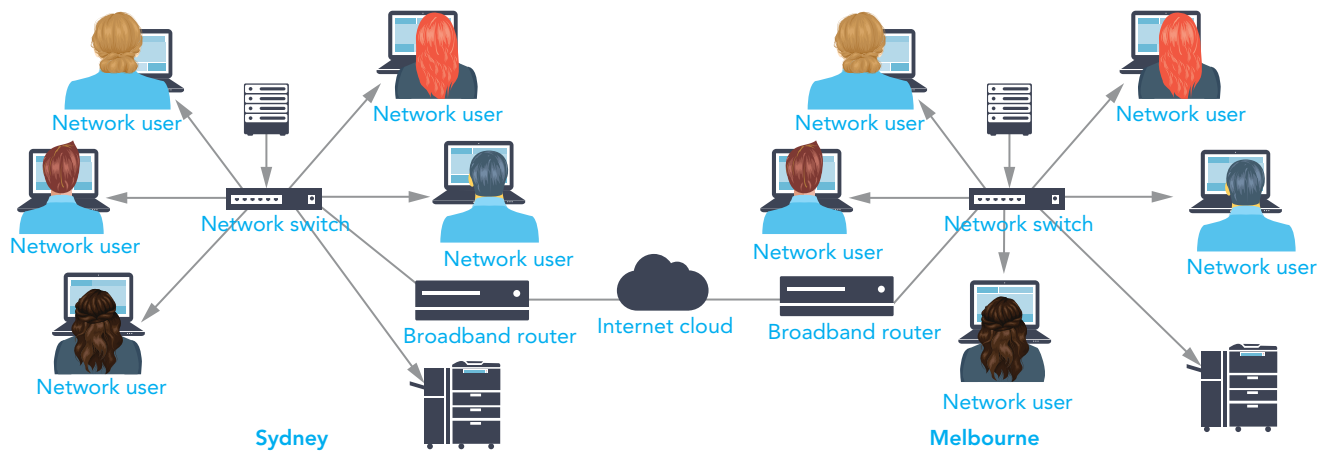


FIGURE 7.5 Two Local Area Networks connected via the internet to form a Wide Area Network

Wireless Personal Area Networks (WPANs)

A **Personal Area Network (PAN)** is a computer network for connecting an individual's devices within a limited range. A **Wireless Personal Area Network (WPAN)** is a type of personal network that uses wireless transmission media to transfer data between a user's personal devices. It allows an individual to connect their devices together and access the internet or a local network using wireless communication techniques. A WPAN may also be referred to as a short-range **wireless network**.

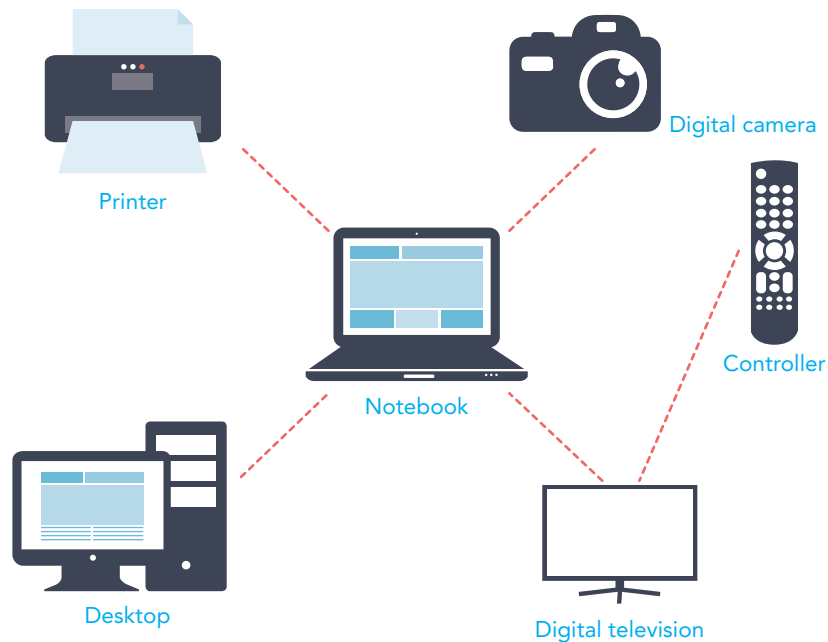


FIGURE 7.6 A Wireless Personal Area Network (WPAN)

Common devices found within a WPAN include desktops, laptops, smart TVs, wearable technology, smartphones and speakers.

A WPAN may use one of the following wireless communication standards:

- 802.11(x)
- Bluetooth
- infra-red (IR)
- satellite
- microwave.

Networking hardware and software

To be able to communicate data between devices on a network, both hardware and software are required. Hardware is the physical components of a network, while software contains instructions to the hardware to enable it to complete particular tasks.

Router

A **router** is a device used to connect multiple networks – several LANs, or a LAN and a WAN (such as the internet). A router will connect LANs and WANs irrespective of the hardware and network communications protocol used in each segment. On the internet or in a large corporate network, for example, routers receive TCP/IP packets, look inside each packet to identify the source and target IP addresses, and then forward these packets as needed to ensure that the data reaches its final destination. Figure 7.7 illustrates how a router can be used to communicate between two LANs operating within one organisation.

While a switch uses a node's Media Access Control (MAC) address to identify the path on which to send the packet, a router uses the IP, IPX or AppleTalk address. The algorithm switches used to decide how to forward packets is different from the one used by routers. Most carriers, such as Vodafone, offer a 4G USB modem stick. Its small size makes it ideal

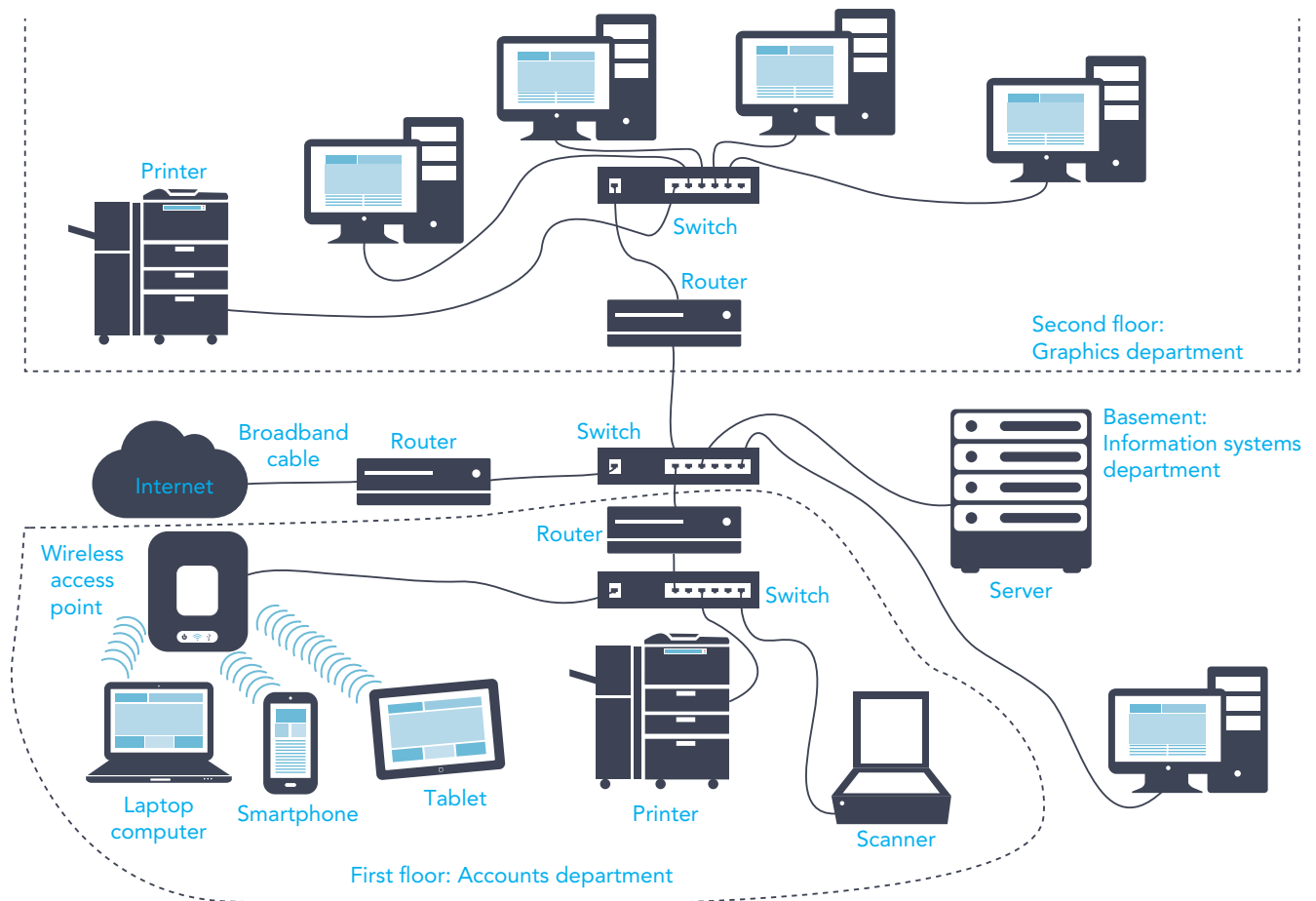


FIGURE 7.7 How a router can connect two LANs and provide internet access to both networks

to use with a mobile device, such as a laptop computer, to connect to the internet using the 4G phone network.

Broadband routers

Routers for home networks (often called **broadband routers**) also can join multiple networks. These routers are designed specifically to join the home (LAN) to the internet (WAN) to share internet connections.

Wireless broadband routers combine the functions of a basic router (connecting the LAN to the internet), a switch (for devices, such as a desktop computer, that are connected by cable), a firewall (security measure) and a **wireless access point** (to allow wireless connectivity). Figure 7.8 shows a broadband router with an antenna for wireless connections, LAN ports for wired connections and a WAN port for connection to the internet.

The type of broadband router used to act as a bridge between a LAN and the internet depends on the type of connection provided by the internet service provider (ISP). A cable connection uses the high bandwidth available through the same broadband connection that delivers information to a television via a provider such as Foxtel. An **asymmetric digital subscriber line (ADSL)** or a **very high bitrate digital subscriber line (VDSL)** connection provides internet access using copper wiring in telephone lines. A cable broadband router or a fibre-optic router is usually sourced from an ISP, since it has to be registered on the ISP's network before it will work.



FIGURE 7.8
Wireless broadband router

Digital subscriber line

Digital subscriber line (DSL) is a type of high-speed internet connection that uses existing telephone lines to transmit digital data, providing internet access without interfering with voice service on the same line.

Asymmetric digital subscriber line (ADSL) routers are designed for ADSL connections that provide higher download speeds than upload speeds. They are suitable for typical home internet usage where downloading activities are more frequent than uploading. ADSL2/ADSL2+ routers are enhanced versions of ADSL that offer improved performance and higher speeds. ADSL2+ routers can provide speeds up to 24 Mbps downstream.

Very high bitrate digital subscriber line (VDSL) routers support VDSL connections that offer faster speeds than ADSL by utilising a wider frequency band. VDSL is suitable for applications that require higher bandwidth, such as streaming high-definition videos and online gaming. VDSL2 routers are an advancement over VDSL and offer even higher speeds and better performance over long distances. They are quite suitable for organisations that have a higher level of internet activities, or for large households with many devices.

Fibre-optic

Fibre-optic internet services are widely available in Australia, particularly in areas served by the National Broadband Network (NBN). The types of fibre connections include Fibre to the Premises (FTTP), Fibre to the Node (FTTN), Fibre to the Building (FTTB) and Fibre to the Curb (FTTC). Fibre-optic routers, also known as fibre routers or NBN routers, are required to connect to these high-speed internet services. A description of each of these types of fibre connections can be seen in Table 7.1.

TABLE 7.1 Different types of fibre connections

Type	Description
FTTP (Fibre to the Premises)	Fibre-optic cable runs directly from the service provider to the user's premises, offering the highest speeds and reliability.
FTTN (Fibre to the Node)	Fibre-optic cable runs to a central node or cabinet in a suburb or neighbourhood, and the final connection to homes is made using existing copper telephone lines.
FTTB (Fibre to the Building)	Fibre-optic cable runs to a building's communications room, and the connection within the building to individual units uses existing copper or ethernet cabling.
FTTC (Fibre to the Curb)	Fibre-optic cable runs to a distribution point or curb near homes, with the final connection to the premises made using existing copper telephone lines.

Switch

A **switch** is a device that provides a connecting point for cables in a LAN.

Network traffic in a LAN typically follows specific paths that connect members of a work group, such as the accounts department of a business. A switch stores the address of every device down each cable connected to it. When a packet arrives, the switch uses simple

logic to detect the destination of the packet and sends the packet only down the cable that contains the addressed device. The result is that the packet reaches its destination quickly and without colliding with packets being sent to other nodes.

As packets are sent straight to the destination device through a switch, the two devices act as if they were directly connected. On a 100 Mbps switch, data can be sent to all nodes simultaneously and uninterrupted at a speed of 100 Mbps.

Encryption of data packets enhances the security of wireless transmissions. Setting up complex passwords for file sharing, routers and access points, as well as using encryption keys that are not obvious, greatly improves security.

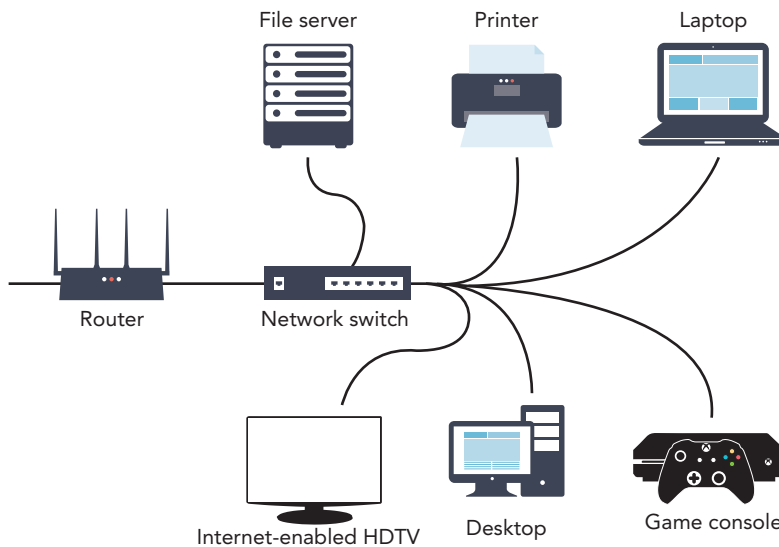


FIGURE 7.9 A switch is used to connect multiple devices.

Defending networks

Networks depend on various hardware and software components to enable secure communication and the safe storage of data and information. These components are essential for defending the network against external threats and ensuring that data is both secure and readily available.

Firewalls serve as the first line of defence in network security, functioning as barriers that monitor and regulate the flow of network traffic between trusted internal networks and untrusted external networks, such as the internet. They assess incoming and outgoing traffic based on established security policies, deciding whether specific data packets can pass through based on factors such as IP addresses, domain names and protocols. Modern firewalls are equipped with inspection capabilities, allowing them to track active connections and make decisions based on traffic context. They also perform deep packet inspection (DPI) for a more detailed analysis, and many are application-aware, offering enhanced security measures against application-based threats.

Data storage and backup systems are crucial for managing and protecting the vast amounts of data generated and used by organisations. These systems not only store data but also ensure its availability and integrity through redundancy techniques, such as RAID configurations, which guard against data loss from hardware failures. They are designed to

be scalable, allowing for increased storage capacity to accommodate growing data needs, and often incorporate automation features for regular data backups, minimising the risk of data loss from corruption or cyber attacks.

On the software side, **network security** encompasses a range of tools aimed at safeguarding the network and its resources. This includes traditional firewall and antivirus software, which protect against unauthorised access and malware, respectively. Additionally, innovative solutions like Darktrace AI leverage artificial intelligence and machine learning to offer advanced threat detection and response. Darktrace AI learns the normal behaviour patterns of a network, enabling it to identify and react to anomalies that may signify security threats, thus offering a proactive approach to network defence.

Systems used to detect intrusions

Systems used to minimise the risk to data and information due to intrusions include intrusion detection systems and intrusion prevention systems.

Intrusion detection systems

An **intrusion detection system (IDS)** is used to monitor network traffic in an attempt to detect intrusions. An IDS can be either hardware-based or software-based. Two types of IDS are host intrusion detection systems and network intrusion detection systems.

A host intrusion detection system (HIDS) is installed on individual devices, including key devices on the network, and monitors and analyses all data packets sent to or from the device. An alert will be sent if suspicious activity is detected. It works by comparing current system files with previous system files. If changes are identified, an alert is sent.

A network intrusion detection system (NIDS) is installed at a strategic point on the network. It functions by monitoring passing network traffic, analysing data packets and checking whether the data packets match any in a library of known intrusion attacks.

An IDS performs a similar function to a firewall but, where a firewall focuses on examining data packets that are passing through an organisation's firewall connection, an IDS focuses on data packets that are travelling within the network.

Intrusion prevention systems

An **intrusion prevention system (IPS)** is often considered an extension of an IDS. Where an IDS focuses on identifying an intrusion and alerting network staff, an intrusion prevention system also attempts to block or stop the threat without the need for human intervention.

An IPS attempts to block intrusions by deleting suspicious data packets, resetting network connections or blocking data packets originating from a particular IP address. An IPS can be configured to follow a set of rules when a particular activity is identified.

Wired, wireless and mobile communications technology

There are a number of communication methods that can be used to transmit data and information over a network. These include wired cabling, wireless transmission and the use of existing mobile (or cellular) networks. The method used to transmit data will depend on

the needs of the users of the network. Each method has strengths and weaknesses in terms of cost, data storage options, data transfer rates, reliability and security.

Wired communication technology

A **wired network** is a network in which the devices are connected using a physical cable.

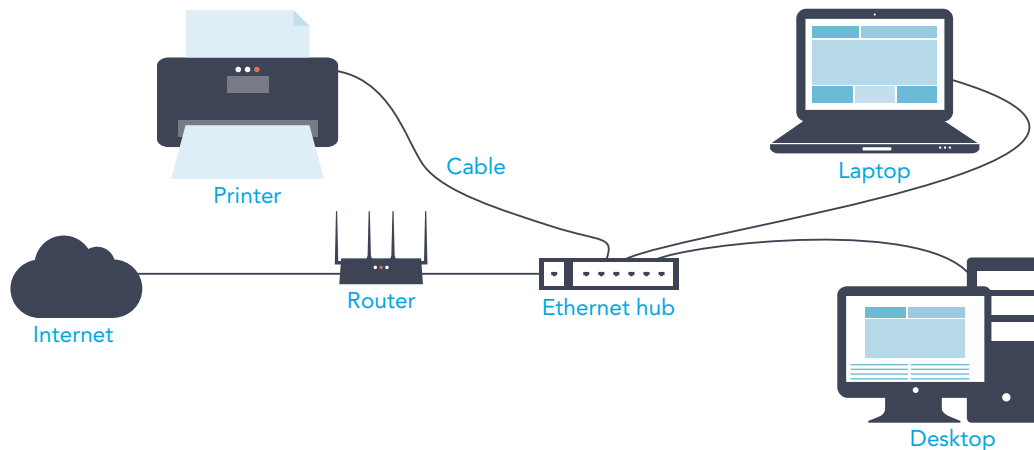


FIGURE 7.10 A wired network

Advantages of wired networks compared to wireless networks include:

- *faster data transfer speeds* – although wireless and mobile network data transmission rates are continually improving, it is still considered that wired networks generate superior speeds (particularly on average)
- *better security* – wired networks are considered more secure than wireless networks. It is much more difficult for a **hacker** to intercept the data in a wired network, particularly a Local Area Network, while there is also less chance that data will be lost due to signal interference
- *more reliable connection* – not only is a wired network considered more reliable in terms of data interference, but wired networks are considered to have more consistent data transfer rates than wireless networks, where the strength of the signal – and hence speed – may vary due to a range of factors.

Disadvantages of wired networks compared to wireless networks include:

- *lack of mobility* – as wired networks generally have fixed data points, which are used to connect devices, there is less flexibility regarding where devices can be used while staying connected to the network. As a result, devices remain stationary, limiting the ability of the user to roam while continuing to use their device
- *installation* – the cost of installing a wired network is considerably more expensive than the cost of a wireless network. The cost, as well as the time and effort, of purchasing and installing cabling is much greater than that of installing a wireless access point
- *maintenance* – from time to time, in a wired network, cabling may stop working correctly, or the organisation may require new data points or rearranging of existing data points. Just as the cost of installation can be significant, maintaining and making changes to an existing wired network are also time-consuming and costly.

Twisted-pair cable

One of the more commonly used transmission media for network cabling and telephone systems is the **twisted-pair cable**. This cable contains one or more twisted-pair wires bundled

Noise in a cable can be caused by cross-talk (electric currents between pairs of wires in the same cable) and by outside electrical fields, such as those created by power lines, motors and radio transmitters. The greater the cable's ability to resist internal and external noise, the longer the cable can be used to connect workstations and resources.

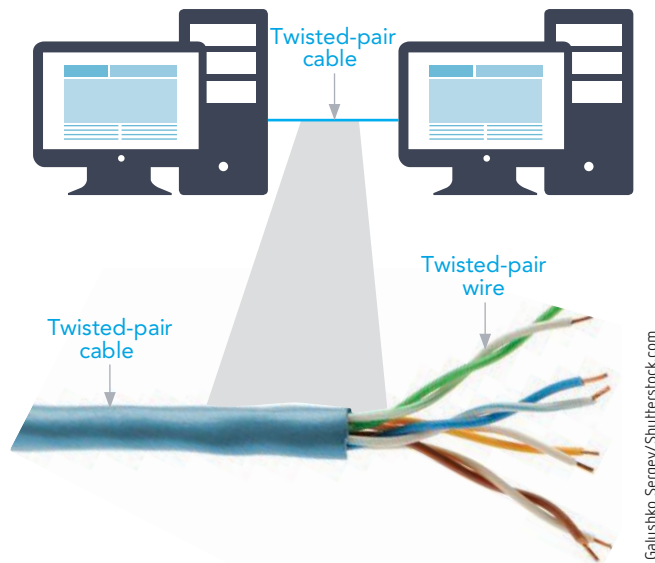


FIGURE 7.11 A twisted-pair cable consists of one or more twisted-pair wires; each twisted-pair wire is usually colour-coded for identification. Telephone networks and LANs often use twisted-pair cables.

together (see Figure 7.11). Each **twisted-pair wire** consists of two separate insulated copper wires that are twisted together. The wires are twisted together to reduce noise. Noise is an electrical disturbance that can degrade communications.

Fibre-optic cable

Fibre-optic cabling is a wired transmission media that contains shards of glass, which reflect pulses of light generated by small lasers or light-emitting diodes (LEDs).

The core of a fibre-optic cable consists of dozens or hundreds of thin strands of glass or plastic that use light to transmit signals. Each strand, called an **optical fibre**, is as thin as a human hair. Inside the fibre-optic cable, an insulating glass cladding and a protective coating surround each optical fibre (Figure 7.12).

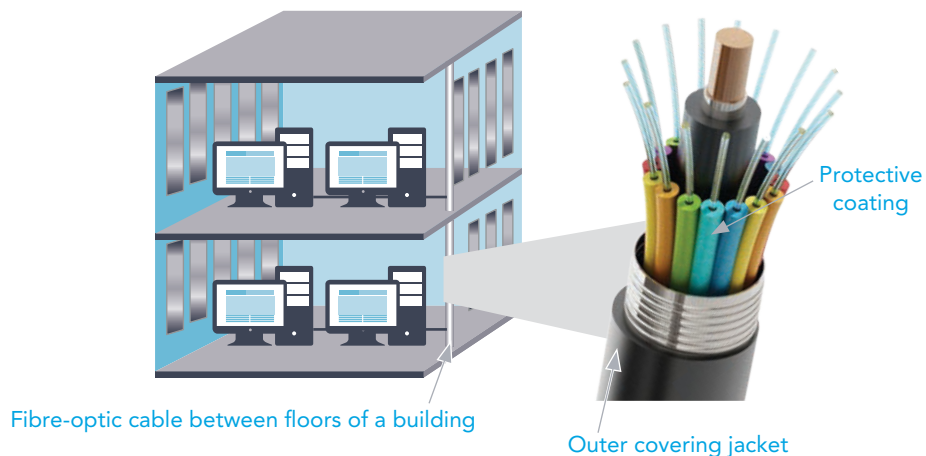


FIGURE 7.12 A fibre-optic cable consists of hair-thin strands of glass or plastic that carry data as pulses of light.

Fibre-optic cables have several advantages over twisted-pair cables. These advantages include:

- the ability to carry significantly more signals than wire cables
- faster data transmission
- less susceptibility to noise (interference) from other devices, such as a copy machine
- better security for signals during transmission as they are less susceptible to noise
- their smaller size (much thinner and lighter).

The material used for fibre-optic cable has optical properties that cause light in the cable to totally internally reflect from the outer surface. This allows all the light to progress down the cable rather than escape to the surrounding air. As fibre-optic cables transmit pulses of light, they have total immunity from electrical noise. Signals can therefore be sent over much larger distances than with twisted-pair cables.

The disadvantages of fibre-optic cable are that it costs much more than twisted-pair cable, and it can be difficult to install and modify. Despite these limitations, many local and long-distance telephone companies and cable television operators are replacing existing telephone and coaxial cables with fibre-optic cables. Many businesses are also using fibre-optic cables in high-traffic networks or as the main cable in a network.

National Broadband Network

The **National Broadband Network (NBN)** is designed to provide the infrastructure to deliver affordable and reliable high-speed internet and telephone access to all Australians. The geography of Australia, and especially its size, means that a variety of technologies are required to deliver the NBN. As originally proposed, the NBN was to deliver communications at a speed of 100 Mbps, with fibre-optic cable connecting homes directly to the internet in a majority of locations. The scheme was modified, and many connections have fibre-optic cable that reaches a node in the street (called the ‘street cabinet’); from there it is split, and hybrid cables or existing copper telephone networks are used to carry signals to homes and businesses. Other types of cabling used include coaxial cabling, fixed wireless and Sky Muster connections.



NBNCO/AAP Image

Households where it is impractical to use fibre-optic cable connect to the NBN via fixed wireless and satellite technologies.

Wireless communication technology

A wireless network is a computer network that uses wireless data transfer between network devices. Wireless networks generally use radio communication to transmit data.

7.5 THINK ABOUT APPLIED COMPUTING

Find out why fibre-optic cabling is not normally used inside a room.

FIGURE 7.13 The rollout of the NBN

Wireless networks can include:

- wireless Local Area Networks (WLANs)
- mobile networks
- satellite networks
- microwave networks.

Wireless transmission media are used when it is inconvenient, impractical or impossible to install cables. The faster speeds afforded by the 802.11ax standard place wireless connections on a par with wired networks in terms of speed. Wireless transmission media used in communications include broadcast radio, cellular radio, microwaves, communications satellites and infra-red.

An 802.11ac wireless router operates on the 5 GHz bandwidth but can still run on the 2.4 GHz network simultaneously. Some vendors may quote a wireless 802.11ac router as a single-speed device operating at 1.75 Gbps. This figure is an amalgamation of the 5 GHz and 2.4 GHz capabilities of the router. That is, it is gained by adding 1.3 Gbps from the 5 GHz connection to 450 Mbps from the 2.4 GHz network.

TABLE 7.2 Transfer rates of various wireless transmission media

Transmission medium	Transfer rate (maximum)
Bluetooth	1–2 Mbps (Bluetooth 2.0 + EDR), up to 3 Mbps (Bluetooth 3.0 + HS), and up to 24 Mbps (Bluetooth 5.0)
NFC	424 Kbps
802.11a	54 Mbps
802.11b	11 Mbps
802.11g	54 Mbps
802.11n (Wi-Fi 4)	108–600 Mbps
802.11ac (Wi-Fi 5)	433 Mbps (single stream), up to 3.5 Gbps (with 8 spatial streams)
802.11ax (Wi-Fi 6)	Up to 9.6 Gbps (theoretical maximum)
Cellular radio	
2G	9.6–14.4 Kbps (GSM), up to 384 Kbps (EDGE)
3G	200 Kbps – 2 Mbps (WCDMA), up to 42 Mbps (HSPA+)
4G	100 Mbps – 1 Gbps (LTE-Advanced)
5G	10–20 Gbps (theoretical maximum)
Microwave radio	150 Mbps (can vary significantly)
Communications satellite	Up to 100 Mbps – 1 Gbps (depending on technology and provider)
Infra-red	115 Kbps – 4 Mbps

Wi-Fi communications

For **Wi-Fi communication** transmissions, you need a transmitter to send the radio signal and a receiver to accept it. To receive the signal, the receiver has an antenna that is located in the range of the signal. Some networks use a transceiver, which both sends and receives signals from wireless devices. Wi-Fi communication is slower and more susceptible to noise than physical transmission media, but it provides flexibility and portability.

Wireless computer network components typically use radio signals in either a 2.4 GHz range or a 5 GHz range. A 5 GHz network can carry more data than a 2.4 GHz network; however, the higher the frequency of a radio signal, the shorter its range. So a 2.4 GHz network covers a much larger range than a 5 GHz network. The higher frequency is not as

good at penetrating solid obstacles such as walls. On the other hand, there are a number of household devices, such as cordless phones, that operate on the 2.4 GHz band and that could interfere with a broadcast transmission at that frequency. The 5 GHz band does not compete with other common household devices. A number of components now come with dual-band capacity to get the best of both worlds.

Bluetooth

Bluetooth uses short-range radio waves to transmit data among Bluetooth-enabled devices. These devices contain a small chip that allows them to communicate with other Bluetooth-enabled devices. Examples of such devices are desktop personal computers, notebook computers, hand-held computers, mobile telephones, fax machines and printers. To communicate with one another, they must be within a specified range (about 10 metres, but the range can be extended to 100 metres with additional equipment). A popular use of Bluetooth is to enable hands-free chatting on mobile phones or to play music through a speaker or in the car. Most new cars are now sold with a built-in Bluetooth station that the user can synchronise with their mobile phone. Bluetooth and Wi-Fi communications technologies use radio signals.

Near-field communication

Near-field communication (NFC) technology is a form of contactless communication between devices such as smartphones and tablets. A user is able to wave their smartphone over an NFC-compatible receiver to send information without needing to touch the devices together or set up a formal connection. In this way, it is possible to pay for goods purchased in stores or pay for a parking meter (the parking meter can even send messages to the smartphone indicating how much time is left).

An unpowered chip, called a tag, can be used with an NFC device, such as a smartphone with NFC capability. The tag uses electromagnetic induction to draw its power from the device that reads it. A smartphone can be paired with an NFC tag, which can be programmed by apps on the phone to automate tasks. For example, tapping a smartphone on a smart tag contained in a poster will transfer information from an embedded chip in the poster onto the smartphone. In this way, a user tapping a movie advertising poster will receive on

Bluetooth gets its name from a legendary Viking king (one of the main developers was the Swedish communications company Ericsson). One advantage of Bluetooth is that it can be used to set up a network on the spur of the moment. A group at a meeting can all network their computers to share files, access data from mobile telephones, and send documents to printers and fax machines, all without cables or additional network interface cards. Of course, all of the devices must be Bluetooth-enabled. Data transmission using Bluetooth is fairly slow compared with other wireless transmission media.

7.6 THINK ABOUT APPLIED COMPUTING

Use the internet to explore creative ways in which people are using NFC tags.



FIGURE 7.14 An example of near-field communication use for short-range contactless communication



FIGURE 7.15 The myki public transportation ticketing system uses near-field communication to read a commuter's details from their smart card or smartphone.

There are five types of NFC tags. Type 1 tags store 96 bytes and operate at 106 Kbps. Type 4 can store up to 32 KB and transmit at 424 Kbps and is typically the fastest. Type 5 tags can provide larger memory than Type 4 depending on the implementation but typically are not faster than Type 4. Type 1 and 2 tags can be written to multiple times, and can use encryption, which permanently locks them so the data cannot be manipulated. Type 3 and 4 tags can be written to once only.

Microwave transmissions are often a viable alternative to cabling where an organisation has premises on different sides of a major road. The cost of digging a tunnel under the road can be expensive and repairs are difficult if there is a break.

FIGURE 7.16

A microwave station is an Earth-based reflective dish that contains the antenna and other equipment necessary for microwave communications; the dish collects the signals and redirects them to the central collector.

An organisation is likely to prefer a licensed microwave link (operating at frequencies between 7 GHz and 42 GHz) if the data being transferred is 'mission critical', if the connection is long-distance or if it operates in a high-density area where interference is a problem at lower frequencies.

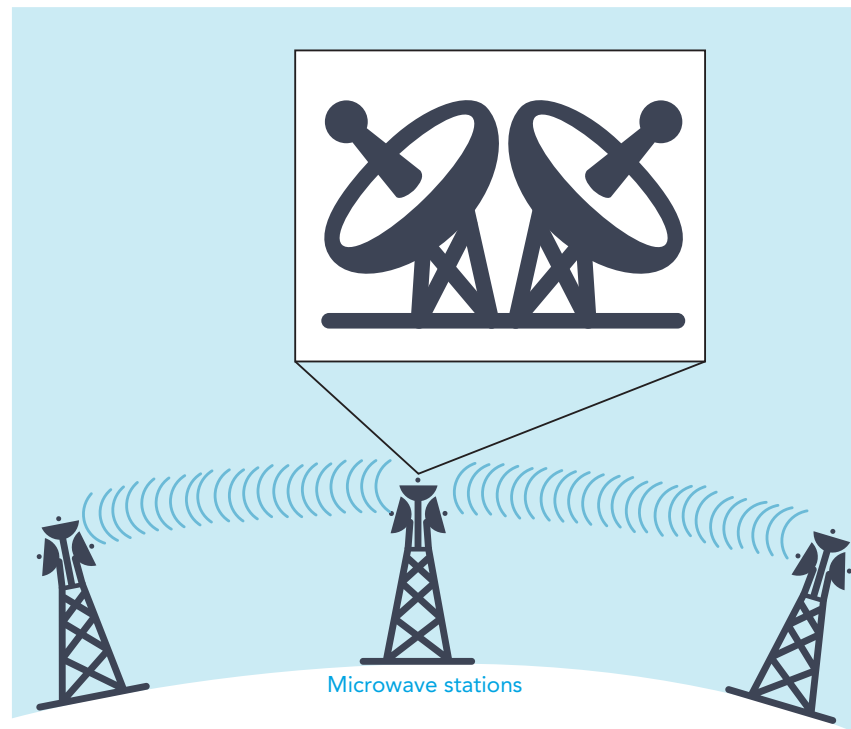
their smartphone comprehensive details about the film, such as session times, biographies of leading actors, reviews and more. Tapping on a menu in a restaurant could load the menu into the phone along with nutritional information and cooking notes.

NFC tags are small and cheap to produce, so are suited for a range of uses involving mobile payments and creative marketing. More interesting uses will emerge as people become more aware of the capabilities of NFC.

PayWave transactions and Victoria's myki transportation system (Figure 7.15) are examples of the use of cards with NFC technology. Waving the card near a card reader allows data to pass from the card to the reader, and hence the transaction is completed with little effort and in almost no time.

Microwaves

Microwaves are radio waves that provide a high-speed signal transmission. Microwave transmission involves sending signals from one **microwave station** to another (Figure 7.16).



Microwaves use line-of-sight transmission, which means that microwaves must transmit in a straight line with no obstructions between microwave antennas. To avoid possible obstructions, such as buildings or mountains, microwave stations often sit on the tops of buildings, towers or mountains.

Electromagnetic radiation, such as light and radio waves, travels almost as fast through the air as it does through a vacuum (about 300 000 km per second). This means that microwave communication is significantly faster than fibre-optic transmissions, which send laser light pulses down glass strands. The glass slows the light beam by 30 to 40 per cent. Microwave transmission is used in environments where installing physical transmission media is difficult or impossible, where the organisation that will be using it occupies a large site and where line-of-sight transmission is available.

Communications satellites

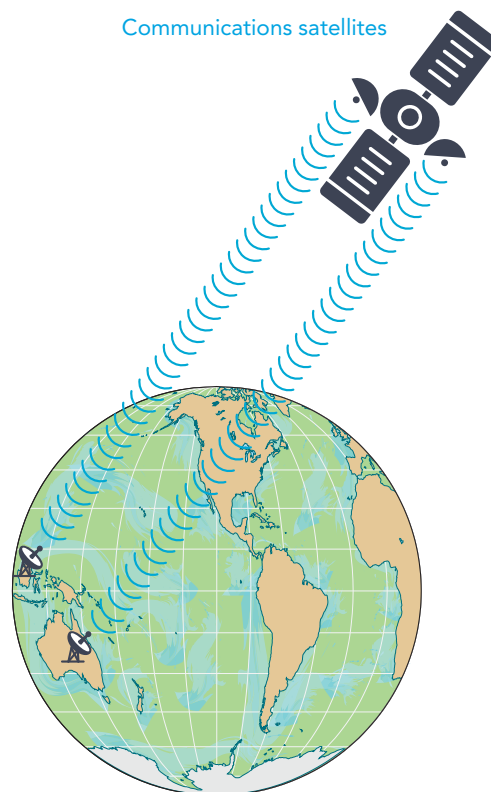
A **communications satellite** is a space station that receives microwave signals from an **Earth-based station**, amplifies (strengthens) the signals and broadcasts the signals back over a wide area to any number of Earth-based stations (Figure 7.17). These Earth-based stations are often microwave stations. Other devices, such as hand-held computers and GPS receivers, can also function as Earth-based stations. Transmission from an Earth-based station to a satellite is an **uplink**. Transmission from a satellite to an Earth-based station is a **downlink**.

Mobile communication technology

A **mobile network** (also known as a cellular network) uses telecommunication networks to allow users to communicate using their mobile device. A mobile network consists of a number of mobile phone towers (or base stations), which send and receive signals. Data coverage of these mobile phone towers often overlaps, allowing users to stay connected to the network as they move around an area.

The range of each mobile phone tower can vary depending on its location and the number of other towers located in the area. In large cities, each tower may only have a short range (e.g. 2 or 3 km), but often these towers overlap with numerous others. In more isolated areas, the tower may have a larger coverage area (e.g. approximately 10 to 50 km, depending on the terrain).

A user's mobile device will connect to the nearest base station. Each base station is then connected to a digital exchange where the communication is sent over a wired network. Mobile networks allow the user to send and receive voice, data, images and text messages. As most (if not all) mobile networks belong to a telecommunications organisation, there is usually a cost associated with transmitting data using these networks.



Although satellite internet connections are more expensive than cable internet, they are still the only high-speed option in remote areas.

7.7 THINK ABOUT APPLIED COMPUTING

There are areas where the coverage of mobile phone towers does not overlap. What are the consequences for a user who is in an area where they are not in reach of a mobile phone tower?

FIGURE 7.17
Communications satellites are placed approximately 37 000 kilometres above Earth's equator.

Strengths and limitations of communication technology

The choice between wired, wireless and mobile communications technologies involves various factors, including cost, data storage options, reliability and security. Each technology has its strengths and limitations, making them suitable for different applications and environments.

TABLE 7.3 Strengths and limitations of communication technology

Criteria	Wired	Wireless	Mobile
Cost	Higher initial set-up cost, due to cabling and installation. Potentially even higher in remote areas due to logistical challenges.	Lower expansion cost compared to wired communication, especially in urban areas. Rural areas might see increased costs for infrastructure to cover larger, less populated areas.	Costs can be high, with variability based on coverage and service providers. Rural and remote areas may face higher costs due to limited competition and infrastructure.
Data storage options	Easily supports large-scale data storage solutions. Remote areas might rely more on local storage due to potentially limited internet connectivity.	Can access remote data storage, but may have speed and reliability issues. Remote data storage access may be affected by connectivity issues in rural and remote areas.	Reliant on cloud storage for expansion. Limited storage on devices may be a constraint in areas with poor mobile data services.
Reliability	Highly reliable with stable connection speeds. Reliability might decrease in remote areas due to infrastructure challenges.	Subject to interference, which can reduce reliability. Interference and range issues can affect rural areas more significantly.	Coverage and reliability vary widely, with urban areas well served and remote areas potentially facing significant coverage gaps.
Security	More secure due to the need for physical connections. Remote areas benefit from reduced access risk.	More vulnerable to unauthorised access and/or eavesdropping.	Faces various security threats, including malware and phishing attacks.
Flexibility	Lacks flexibility as adding or relocating devices requires physical changes. This might impact remote areas more due to the cost and difficulty of installing such infrastructure.	Very flexible and mobile within the range of the network. In remote areas, coverage limitations reduce this advantage.	Offers the highest level of mobility, crucial for connectivity in remote and rural areas despite potential coverage challenges.

Technical underpinnings of networks

Intranets

An **intranet** is a private network accessible only to users within an organisation.

An organisation's internal intranet is unlike the internet in that a wide range of information and services that are available on it, including calendars, policies, procedure manuals and technical support files, are unavailable to the public. An intranet also allows access to documents required by users within the organisation.

Many intranets have search engines, user profiles, blogs, notifications and event planning functionality.

An intranet is generally a website-based information system, made up of a number of HTML pages that are stored on a web server (similar to any other website). Users of the intranet access the information using a web browser, such as Chrome, Safari or Firefox.

An intranet will often include a connection to the internet, allowing employees access to information from the web. Intranet pages will often include links to internet sites with information relevant to the organisation.

An intranet provides the following efficiencies:

- It facilitates communication by allowing employees to work in groups.
- Users can access information faster since data does not need to pass through a router, and loading graphics and images becomes much quicker than on the internet.
- An intranet reduces paper waste because companies are able to move their documents and processes onto the intranet.
- An intranet improves ease of use. Point-and-click technology allows easy access to hyperlinked company documents.

The effectiveness benefits of an intranet include the following:

- An intranet allows restricted access to company information.
- An intranet provides dynamic information. Company documents stored on an intranet can be updated more easily and quickly than hard-copy versions.
- An intranet allows connection across different network platforms. Users of complex networks that employ a number of different operating systems – such as Microsoft Windows, Macintosh and UNIX – are able to communicate easily within an intranet using their browser software. The HTML code used by the web pages is universal across all platforms.
- An intranet makes data more accessible. A user who is authorised to connect to the internal network can access the information stored on an intranet from anywhere in the world via the internet.
- An intranet supports the ability to communicate using audio and video files rather than just text and graphic images.

Internet

The **internet** is a worldwide network in which billions of devices are connected together for the purposes of communication and data exchange.

The internet uses common communication protocols to link the network devices with each other. Two major concepts that allow devices to communicate on the internet are IP addresses and the protocol TCP/IP.

IP addresses

Every device that is connected to the internet must have a unique address. This address is called the IP address. The IP addressing standard is four numbers, each between 0 and 255, separated by full stops.

IP stands for Internet Protocol. In computing, a protocol is a set of rules. The Internet Protocol is a set of rules that are followed to allow devices to communicate over the internet.

An IP address is similar to a street address for a house. When we post a letter, the receiver's address is written on the front of the letter. This allows the mail service to deliver the letter to the correct address. Also, the sender's address is often written on the letter, so the receiver knows who sent the letter. On the internet, an IP address plays a similar role.

There are two types of IP addresses: private IP addresses and public IP addresses.

A private IP address is assigned by the user. There are three blocks of IP addresses that are dedicated for private use, usually ranging between 192.168.1.1 to 192.168.1.254. A private IP address is not globally unique as anyone can use the IP address within their own private network.

7.8 THINK ABOUT APPLIED COMPUTING

Open the command prompt on your device and input the command **ipconfig** to display your private IP address.

7.9 THINK ABOUT APPLIED COMPUTING

Use a website such as What Is My IP? to identify your public IP address.



Weblink
What Is My IP?

The protocol IPv6 offers a larger range of IP addresses than earlier versions of the protocol.

```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.17763.437]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\asul0002>ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

    Connection-specific DNS Suffix  . : esol.monash.edu.au
    IPv6 Address. . . . . : 2001:388:608c:4ca5:2af1:eff:fe11:de6d
    Temporary IPv6 Address. . . . . : 2001:388:608c:4ca5:2461:d829:b7:1350
    Link-local IPv6 Address . . . . . : fe80::2af1:eff:fe11:de6d%10
    IPv4 Address. . . . . : 130.194.105.247
    Subnet Mask . . . . . : 255.255.255.128
    Default Gateway . . . . . : fe80::21c:eff:feeb:fac3%10
                               fe80::214:a8ff:fee2:b1c2%10
                               130.194.105.254

Wireless LAN adapter Local Area Connection* 1:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 2:

    Media State . . . . . : Media disconnected
    Connection-specific DNS Suffix  . :

```

FIGURE 7.18

Screenshot of console window showing a private IP address

A public IP address is a globally unique IP address that is assigned by an ISP. The ISP assigns a public IP address to a network device connected to the internet either dynamically or statically.

TCP/IP

The **Transmission Control Protocol/Internet Protocol (TCP/IP)** is a set of rules that allow communication between two networked devices.

TCP/IP is a combination of two protocols (TCP and IP). It ensures that the messages travelling over the internet reach the destination IP address. TCP/IP can also be used on Local Area Networks (LANs).

Transmission Control Protocol

When data is sent over the internet, TCP:

- converts the data into packets
- sends each packet towards the destination
- arranges the packets in the correct order when they arrive
- reassembles the packets back into the correct format at the destination.

TCP is also responsible for resending any packets lost during the communication process.

Internet Protocol

IP is responsible for addresses and the routing of the packets to the correct destination. It uses packet switching to do this.

Packet switching

When data is sent across the internet, TCP breaks it down into small packets. When IP directs each packet across the internet, it is possible that each packet may travel by a different route to the destination.

As each individual packet is sent, IP identifies the most efficient route to the destination at that time. As the next packet is sent, the most efficient route is recalculated, which improves the data transfer speed of sending data over the internet.

Accessing websites

A website consists of a number of web pages that can be accessed over the internet by entering the site's domain name or internet address. All the resources of the website (e.g. pages, files, multimedia content) are stored on a web server. Behind each domain name is an IP address that directs the request to the location of the web browser where the resources are stored.

Virtual private network (VPN)

A **virtual private network (VPN)** allows a user to use public networks such as the internet to send and receive data and information as if they were using a private network, such as an intranet.

Virtual private networks were developed to allow users to access an organisation's applications and resources securely from remote locations. Security is maintained by employing protection strategies such as requiring users to authenticate their identity with passwords or similar techniques when they attempt to log in to the VPN.

A VPN can also be used to avoid geo-restrictions and censorship, and to protect users' anonymity and conceal their location while they are surfing the internet.

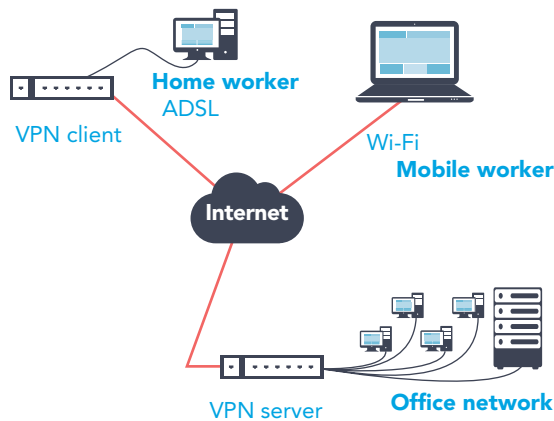


FIGURE 7.19 A virtual private network allows remote workers to connect to the office network.

Network diagrams

A common design tool used to represent the appearance of a network is a layout diagram. A layout diagram can be used to create a network diagram that allows a topology view of a network.

A **network diagram** is a visual representation of a network that shows the nodes that make up a network and how they interact (Figure 7.20). This can include devices such as desktop, laptops and mobile devices, network hardware including servers, routers, switches and modems, and communication methods including wired, wireless and mobile communication. Security controls used, including authentication techniques, firewalls and system protection software and protocols, can also be included or annotated on the diagram.

Note that Figure 7.20 is not based on the plans of the building. This would make the diagram cumbersome and difficult to read. The physical buildings are not important in a network diagram, but it is necessary to identify work areas, such as the dispatch department and the marketing department in this particular example.

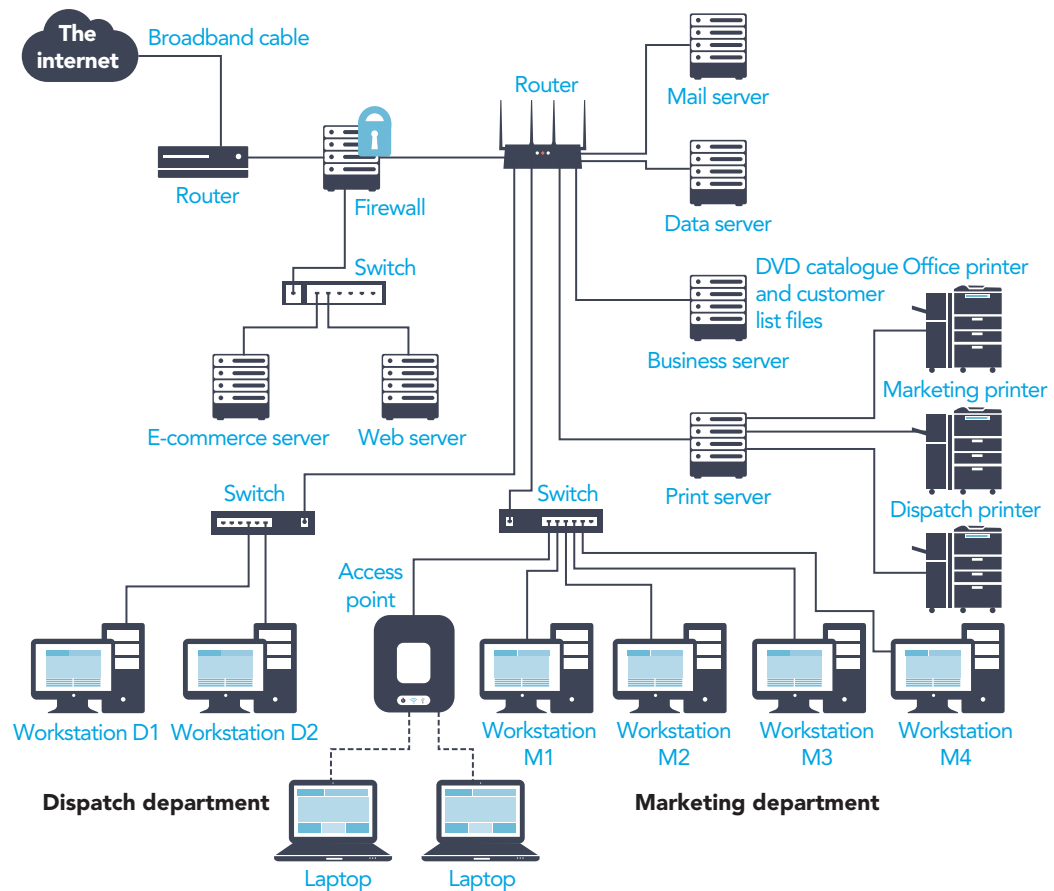


FIGURE 7.20 Network diagram

Artificial intelligence and network security

Artificial intelligence (AI) in network security has revolutionised how organisations protect their digital environments. Specifically, the use of machine learning (ML) algorithms to analyse network traffic patterns and the provision of real-time monitoring and notifications are at the forefront of this evolution.

Machine learning (ML) algorithms

ML algorithms are a subset of AI that learn from data, improving their accuracy over time without being explicitly programmed for each new threat. In the context of network security, ML algorithms analyse vast amounts of network traffic data to identify patterns and behaviours typical of both normal operations and potential security threats. By continuously learning from the data flowing through the network, these algorithms can differentiate between authentic activities and malicious ones, such as unusual data transfers or attempts to access restricted areas of the network.

This capability is particularly valuable for identifying sophisticated cyber threats that might evade traditional security measures. For example, ML algorithms can detect anomalies that signal the presence of advanced persistent threats (APTs), which are stealthy and continuous hacking processes, often orchestrated by a person or a group targeting a specific entity. These

algorithms are trained on both historical and real-time data, enabling them to recognise the evolving tactics, techniques and procedures used by cyber attackers.

Real-time monitoring and notification

Real-time monitoring and notification systems powered by AI provide immediate alerts about potential security incidents, allowing for swift response and mitigation. These systems continuously scan network traffic and analyse it using machine learning algorithms to detect signs of security breaches or vulnerabilities. When a potential threat is identified, the system automatically generates an alert, providing detailed information about the nature of the threat, its location within the network and recommended actions to contain and neutralise it.

This real-time capability ensures that security teams are not just reacting to breaches after they have occurred but are proactively monitoring for signs of compromise. It significantly reduces the time between the initial breach and its detection, a critical factor in minimising the impact of cyber attacks. Moreover, real-time alerts can be prioritised based on the severity and potential impact of the threat, enabling security teams to focus their efforts on the most critical issues.

Risks of using networks in a global environment

Operating in a global environment has various risks associated with the use of networks. These challenges encompass cyber security threats, data privacy concerns, the complexity of legal compliance and the risk of unauthorised network access.

Cyber security threats

Global networks increase exposure to a wide array of cyber security threats, including malware and ransomware. The diverse and often sophisticated nature of these threats can lead to data breaches, financial loss and damage to an organisation's reputation. Operating globally means facing attackers from different countries, each with unique tactics and levels of sophistication, making the cyber security landscape more challenging to manage.

Data privacy

Data privacy becomes significantly more complex in a global environment due to varying and sometimes conflicting data protection laws across countries. Organisations must ensure the privacy of personal and sensitive information in accordance with each country's regulations where they operate. This includes compliance with regulations such as the European Union's General Data Protection Regulation (GDPR), which has strict requirements for data handling and consumer rights to data privacy. Non-compliance can result in large fines and legal action.

Legal compliance

Navigating the legal requirements of multiple countries adds layers of complexity to global network operations. Beyond data privacy laws, organisations must comply with regulations regarding cyber security practices, data sovereignty (where data must be stored within the country of origin) and cross-border data transfers. Each jurisdiction may have its own set of

rules and standards that organisations must adhere to, requiring a thorough understanding of local laws and potentially necessitating different compliance strategies for different regions.

Unauthorised network access

The risk of unauthorised access to networks is magnified in a global setting, where networks span across borders and may be subject to different levels of security controls and oversight. The more extensive the network, the greater the number of potential vulnerabilities and entry points for attackers. Unauthorised access can lead to data theft, espionage, sabotage and the introduction of malware into the network. The challenge is further compounded by the need to manage access for a globally distributed workforce, necessitating robust identity and access management solutions.

Reducing risks and mitigating threats

Reducing risks and mitigating threats to networks involve several key practices that target different aspects of network and information security, such as application of firmware, multi-factor authentication, backup strategies and software malware updates. By addressing vulnerabilities at both the hardware and software levels, enforcing strong authentication measures, ensuring data resilience through backups and maintaining up-to-date defences against malware, organisations can significantly reduce their risk profile and enhance their overall security.

Application of firmware updates

Firmware is the low-level software programmed into the hardware components of network devices, such as routers, switches and firewalls. Manufacturers release firmware updates to address vulnerabilities, add new features or improve performance.

Applying firmware updates is crucial because vulnerabilities in firmware can be exploited by attackers to gain unauthorised access, disrupt services or launch attacks against other devices on the network. Regularly updating firmware helps close these security gaps, ensuring that the underlying hardware of the network infrastructure remains secure.

Multi-factor authentication (MFA)

MFA requires users to provide two or more verification factors to gain access to a resource, such as a network or system. These factors can include something the user knows (password), something the user has (security token, smartphone app), and something the user is (biometric verification).

MFA significantly enhances security by adding layers of defence, making it much harder for attackers to gain unauthorised access even if they manage to compromise one form of identification, such as a password. This is particularly important in protecting sensitive systems and data from unauthorised access and breaches.

Backup strategies

Backup strategies involve creating copies of data and systems that can be restored in the event of data loss, corruption or a cyber attack, such as ransomware. Effective strategies include

regular backups, off-site storage and testing backups to ensure that they can be successfully restored.

Backup strategies are a critical component of disaster recovery and business continuity planning. They ensure that an organisation can quickly recover from an incident without significant data loss or downtime, thus mitigating the impact of cyber attacks and other data loss scenarios.

Software and malware updates

This practice involves regularly updating software applications and operating systems with patches released by vendors to address security vulnerabilities, bugs and performance issues. It also includes updating antivirus and anti-malware software with the latest definitions and signatures.

Software and malware updates are essential for protecting networks from known vulnerabilities and threats. Cyber criminals often exploit outdated software to launch their attacks. By keeping software up to date, organisations can protect themselves against these exploits. Similarly, updated anti-malware tools are better equipped to detect and remove the latest malware threats, providing an additional layer of defence against malicious software.

Cryptographic techniques to protect networks

It is important to use a variety of security measures to prevent unwanted access to networks and data.

Ciphers

Ciphers are used to protect data and networks because they provide a fundamental layer of security through the encryption of information. Encryption is the process of converting plain, readable data (plain text) into an unreadable format (cipher text) using a cipher, which is a specific algorithmic transformation.

Ciphers ensure that sensitive information remains confidential. Only individuals or systems with the correct decryption key can revert the cipher text back to its original plain text form. This means that even if data is intercepted during transmission over a network or accessed without authorisation, it remains indecipherable and useless to the attacker.

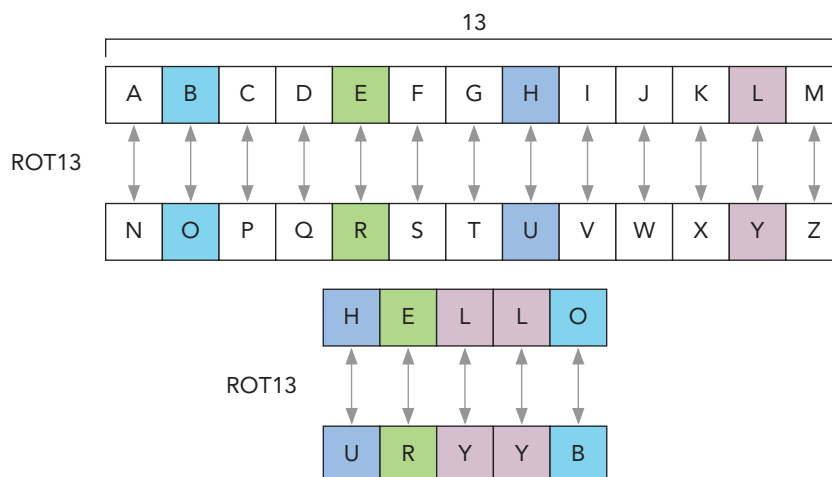
Ciphers can also be used to verify the integrity of data. By encrypting and then decrypting data, we can ensure that it has not been altered in transit. This is often combined with techniques such as hashing and digital signatures to provide a stronger guarantee that data received is exactly the same as the data sent, safeguarding against tampering.

Three ciphers that are considered in Applied Computing are **Caesar cipher**, **polyalphabetic cipher** and **Vigenère cipher**.

Caesar cipher

The Caesar cipher is a type of substitution cipher in which each letter in the plain text is shifted a certain number of places down or up the alphabet. For example, with a shift of 3, A would be replaced by D, B would become E, and so on. A basic example of a Caesar cipher using a shift of 13 is known as ROT13 and is shown in Figure 7.21.

FIGURE 7.21 A basic example of a method used to encrypt data. The alphabet is broken into two halves. Each letter is paired with another letter. When data is encrypted, each letter in the data is replaced by its pair. The data 'HELLO', once encrypted, would be transmitted or saved as 'URYYB'.



Polyalphabetic cipher

A polyalphabetic cipher uses multiple substitution alphabets to encrypt the message, making it more secure against frequency analysis. The fundamental idea behind this approach is to obscure the relationship between the plain text and cipher text characters, making frequency analysis, which is a common method for breaking simple substitution ciphers, much more difficult.

In a polyalphabetic cipher, the basic principle involves shifting the alphabet used for encryption based on a certain rule or key, which dictates the sequence of alphabets to apply. This sequence repeats throughout the message. Each alphabet shift corresponds to a different substitution cipher, hence the term 'polyalphabetic' (meaning 'many alphabets').

The encryption process begins with defining a key, which could be a word, a number or a sequence of characters. This key controls how the alphabets are cycled and applied to the plain text. For each letter of the plain text, an alphabet shift is applied based on the key. This shift might move 'A' to 'B' (a shift of 1), to 'C' (a shift of 2), and so on. The specific shift changes according to the key, and unlike monoalphabetic ciphers, which use a single shift for the entire message, polyalphabetic ciphers vary this shift, making the encryption more complex.

When the cipher encrypts, the plain text letter is replaced with the letter from the shifted alphabet, based on the current position within the key sequence. Once the end of the key is reached, the sequence starts again from the beginning of the key, continuing this pattern until the entire message is encrypted.



CASE STUDY

Polyalphabetic cipher: A sample cipher

Let's create a cipher using a key of three numbers to create a sequence of different Caesar shifts for each letter in the plain text.

Key: 314 (This key means the first letter will be shifted by 3, the second by 1, the third by 4, and this pattern repeats for the rest of the message.)

We use the English alphabet, and a shift means moving letters down the alphabet by the specified number of positions. After 'Z', we loop back to 'A'.

Plain text message: POLYALPHABETIC

Encryption process

P is the 1st letter (shift by 3):
 O is the 2nd letter (shift by 1):
 L is the 3rd letter (shift by 3):
 Y is the 4th letter (back to the first shift, 3):
 A is the 5th letter (shift by 1):
 L is the 6th letter (shift by 4):
 P is the 7th letter (shift by 3):
 H is the 8th letter (shift by 1):
 A is the 9th letter (shift by 4):
 B is the 10th letter (shift by 3):
 E is the 11th letter (shift by 1):
 T is the 12th letter (shift by 4):
 I is the 13th letter (shift by 3):
 C is the 14th letter (shift by 1):

P → Q → R → S
 O → P
 L → M → N → O → P
 Y → Z → A → B
 A → B
 L → M → N → O → P
 P → Q → R → S
 H → I
 A → B → C → D → E
 B → C → D → E
 E → F
 T → U → V → W → X
 I → J → K → L
 C → D

Cipher text: SPPBBPSIEEFXLD

Vigenère cipher

The Vigenère cipher is a polyalphabetic cipher method of encrypting alphabetic text. In this application of an alphabetic cipher, the key is a word or a phrase. Each letter of the key refers to a shift used to encrypt the plain text letter at the same position.

Polyalphabetic cipher: A Vigenère cipher

Key: PIE

Plain text message: POLYALPHABETIC

Encryption process

Each letter of the plain text is encrypted with a letter from the keyword. The shift for each letter is determined by the position of the keyword letter in the alphabet (P = 15, I = 8, E = 4), where A = 0.

P (with P shift, 15 positions):	P → E
O (with I shift, 8 positions):	O → W
L (with E shift, 4 positions):	L → P
Y (with P shift, 15 positions):	Y → N
A (with I shift, 8 positions):	A → I
L (with E shift, 4 positions):	L → P
P (with P shift, 15 positions):	P → E
H (with I shift, 8 positions):	H → P
A (with E shift, 4 positions):	A → E
B (with P shift, 15 positions):	B → Q
E (with I shift, 8 positions):	E → M
T (with E shift, 4 positions):	T → X
I (with P shift, 15 positions):	I → X
C (with I shift, 8 positions):	C → K

Cipher text: EWPNIPEPEQMXXK

**CASE
STUDY**



THINK ABOUT APPLIED COMPUTING

7.10

How would the data 'YESTERDAY' be stored after it was encrypted using the Vigenère cipher?

Encryption

Encryption is a security control that is used when communicating data and that can also be used to protect stored data. **Encryption** involves encoding or changing data so that it can only be decrypted by the receiver if they have the correct **key**. An unauthorised user might gain access to the data and information, but without the correct key they would not be able to understand it.

A credit card number used to purchase goods online is an example of data that would benefit from encryption.

Symmetric and asymmetric keys

To be able to encrypt (and decrypt) data, a key is needed. A key can be used when there is data that requires protection. A key is a series of bits (0s or 1s in binary code) that is used to apply an algorithm to the data packet to scramble (encrypt) and unscramble (decrypt) the data. In other words, a key is used to 'lock' and 'unlock' the data. Encryption software is used to create a key.

A key is a long random number, such as:

```
4356 0241 00C9 18DF CF8D EB2D EFD5 FD37 89B9 E069 EA97 FC20 5E35 F567
RE31 C4FB C6E4 4811 7D86 BC8F BAFA 362F 922B F05R 2F88 C133 2654 C0DR 2881
D673 BN2B 4673 C266 E2CD CB02 0301 3451
```

Common key sizes include 128, 192 and 256 bits. The larger the key, the more secure the encryption.

Two types of encryption are symmetric and asymmetric encryption. **Symmetric encryption** requires both the sender and the receiver to use the same private key. A **private key** is a key that is only known by a user who has the key installed on their device.

The sender uses the key to encrypt the data packet, and the receiver needs the same key to decrypt that data packet. A private key is also referred to as a secret key.

AES (Advanced Encryption Standard) is a widely used symmetric encryption method. It works by taking data and a secret key, mixing them together through a series of complex steps called rounds and producing encrypted data that looks random. The number of rounds depends on the key length: 10 rounds for a 128-bit key, 12 for a 192-bit key, and 14 for a 256-bit key.

During encryption, AES substitutes bytes, shifts rows of data, mixes columns and combines the data with parts of the key. To decrypt the data, the process is reversed using the same key, returning the data to its original form.

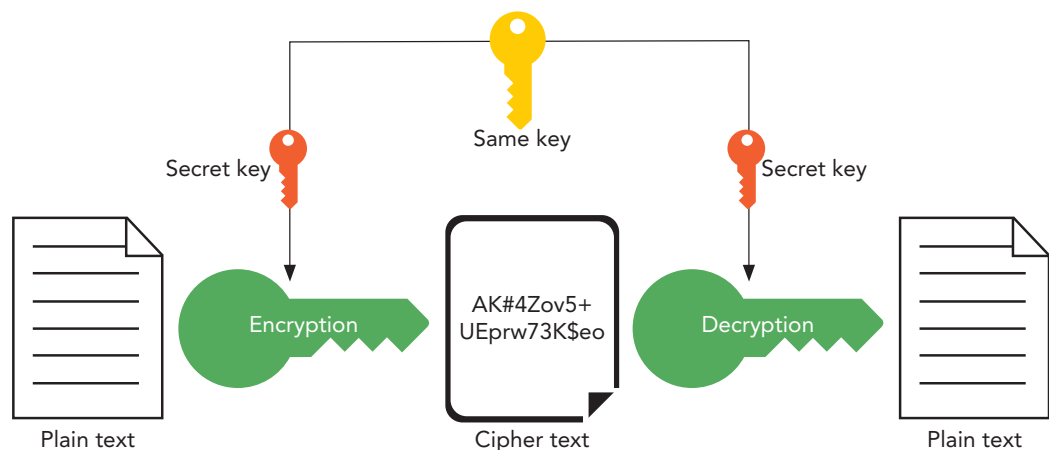


FIGURE 7.22
Visualisation of symmetric encryption

Asymmetric encryption requires the user to install two keys on their device: a private and a public key. A **public key** is a key that is made available to other devices and is used to encrypt data packets that are being sent to the user, while the private key is used on the receiving device to decrypt the data.

The algorithms used by the two keys are related, so that data encrypted by a public key can only be decrypted by a private key.

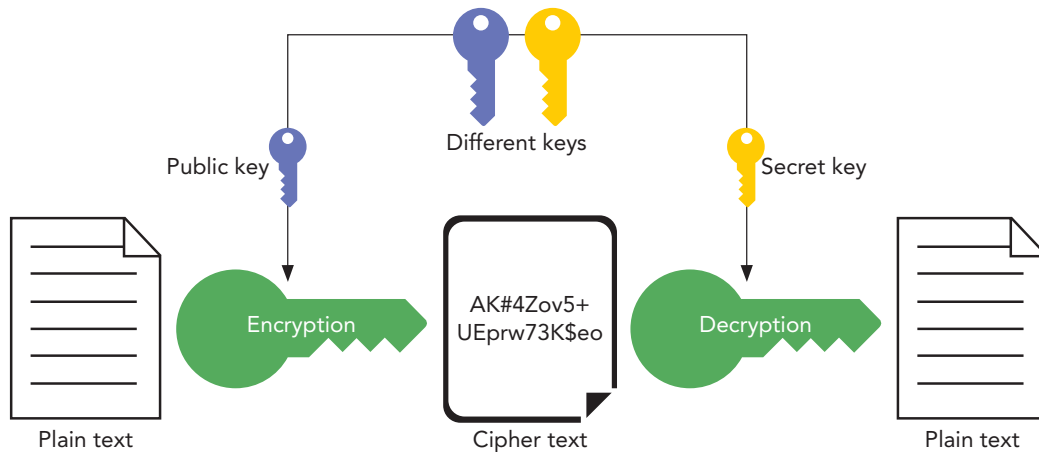


FIGURE 7.23
Visualisation of
asymmetric encryption

In Applied Computing, there are three main types of asymmetric encryption that are required knowledge: RSA, ECDH and ECDSA.

RSA (Rivest-Shamir-Adleman) encryption was one of the first public-key cryptosystems, and it is widely used for secure data transmission. It relies on the mathematical properties of large prime numbers. RSA is also used for digital signatures, where the private key is used to sign a message, and the public key is used to verify the signature, ensuring the message's authenticity and integrity.

ECDH (Elliptic-Curve Diffie-Hellman) is an encryption method used to securely exchange cryptographic keys over a public channel. It uses the mathematics of elliptic curves to create public and private keys, which can then be used to encrypt and decrypt communications.

ECDSA (Elliptic Curve Digital Signature Algorithm) is used for digital signatures, ensuring data integrity and authenticity. Like ECDH, it also uses elliptic curves to create a pair of keys: a private key for signing data and a public key for verifying the signature. When a message is signed with a private key, the signature can be verified by anyone with the corresponding public key, proving that the message has not been altered and confirming the identity of the signer. This method is more efficient than older algorithms like RSA, providing the same level of security with smaller key sizes.

Security threats on networks

A **threat** to a network is anything that has the potential to cause harm to data and information stored and communicated between information systems. A threat may or may not occur, but it has the potential to result in the loss, theft or damage of data and information. A number of measures can be used to minimise the chances of a security threat.

Denial-of-service attacks

A **denial-of-service (DoS) attack** occurs when hackers attempt to prevent legitimate users from accessing a service from an information system or network.

A DoS attack usually takes the form of a very large number of messages, which ask the network or server to authenticate requests, but that appear to be sent from non-existent addresses. When the network or server responds by sending authentication approval, it is unable to find the invalid return addresses. It keeps trying, causing the server to wait before closing the connection.

When the server finally closes the connection, the attacker sends more authentication requests, again with invalid return addresses. The network or server is kept busy, trying to respond but unable to find the return addresses. This prevents legitimate users from accessing the services.

Hackers may also send fake data packets to every computer on a network instead of only one, or may carry out a SYN (synchronisation) flood, in which a server connection request is sent to every open port but the acknowledgement process or 'handshake' is never completed, which leaves no ports open for legitimate users.

Distributed denial-of-service (DDoS) attacks (Figure 7.24) are similar to DoS attacks except that they originate from many sources instead of one. This increases the effectiveness of the attack as the amount of traffic on the network is increased significantly. It also makes it much more difficult to shut down the attack and to identify its source.

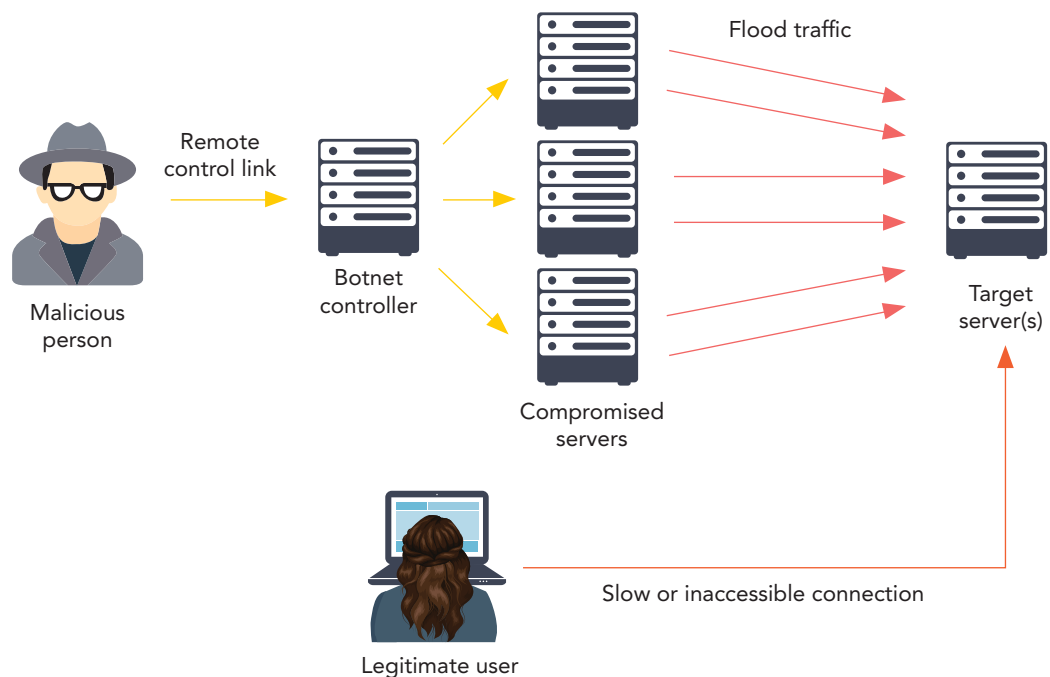


FIGURE 7.24 Visual representation of a DDoS attack

THINK ABOUT APPLIED COMPUTING

7.11

How does your school/institution manage user access once students complete their VCE studies?

Improper credential management

Improper credential management can significantly undermine the security of data and information systems.

The lack of strong password policies can lead to the use of weak, easily guessable passwords. This includes passwords that are short, common or lack a mix of characters (letters, numbers, symbols). Weak passwords are more susceptible to brute-force attacks, where attackers use automated tools to guess passwords.

When users reuse passwords across multiple accounts and services, a breach in one system can compromise all other accounts using the same password. This practice amplifies the impact of data breaches, allowing attackers to gain unauthorised access to a wide range of resources.

Failing to implement or enforce the use of multi-factor authentication (MFA) leaves systems vulnerable to unauthorised access. MFA adds an additional layer of security by requiring a second form of verification beyond just a password, such as a code sent to a mobile device or a fingerprint scan. Without MFA, compromised credentials can easily lead to unauthorised access.

Storing or transmitting credentials in unencrypted formats exposes them to interception and misuse. Proper credential management requires encrypting passwords and other sensitive authentication data both when stored and when transmitted in order to protect against eavesdropping and data theft.

The lack of secure and efficient processes for resetting and revoking credentials can lead to security gaps. For example, if an employee leaves an organisation and their access is not promptly revoked, or if a compromised account's password is not quickly reset, attackers can exploit these oversights to gain or maintain unauthorised access.

Malicious software

Malware (short for 'malicious software') is software that is designed to damage, disrupt or gain unauthorised access to an information system. The term 'malware' is used to group together a range of software threats to data and information.

Each type of malware functions differently. Below is a brief description of the functionality of some common forms of malware.

Viruses

A virus is a software application that is written to cause some type of negative effect on a device or network. When a virus is executed, it normally duplicates itself in a range of locations on a device. A virus may be designed to damage, steal, modify or corrupt data.

Viruses work by attaching themselves to legitimate programs or files and, when these infected files or programs are executed, then activating and beginning malicious activity. This can include data corruption, stealing information, disrupting services or allowing unauthorised access to the infected systems.

The infection process starts when a virus attaches its code to a host file or program, often through email attachments, downloaded files or by exploiting vulnerabilities in software. Once the infected file is executed, the virus replicates by attaching its code to other files or programs on the system. It can spread to other systems through networks, shared drives or removable media like USBs. The virus remains dormant until the infected file or program is run. Some viruses have a specific trigger condition, such as a particular date or user action, that activates their payload.

Upon activation, the virus executes its payload, which can range from displaying annoying messages to more harmful actions such as deleting files, corrupting data or providing remote access to attackers. The virus continues to spread by infecting more files and systems, typically using networks and other communication channels.

Computer viruses rely on spreading to multiple hosts to cause widespread harm. They exploit the trust users have in receiving files or emails from known sources, which increases the importance of users putting in place good cyber security measures, such as not opening

suspicious attachments, keeping software updated and using antivirus software to protect against these threats.

Worms

Unlike a virus, a worm is software that, once installed, self-replicates. Worms exploit vulnerabilities in network protocols or software to gain access to new systems. Once inside a system, a worm scans the network for other vulnerable systems and continues to replicate and spread on its own. Worms operate independently by exploiting security weaknesses, allowing them to move rapidly from one system to another.

Worms can cause significant damage quickly by consuming network bandwidth, overloading servers, and potentially carrying harmful payloads that can delete files, steal data or create back doors for further exploitation. The rapid and autonomous nature of worms makes them particularly dangerous and disruptive.

To protect against worms, it is essential to keep software and systems updated with the latest security patches, monitor network traffic for unusual activity and use firewalls and antivirus programs to detect and block potential worm activity.

Spyware

Spyware is malicious software that collects data about a user's activity and sends that data to another location without the user's knowledge. It infiltrates systems through deceptive methods such as malicious websites or email attachments. Once installed, spyware embeds itself in the system, often running in the background to avoid detection.

Spyware comes in various forms, each serving different malicious purposes. Keyloggers record keystrokes to capture sensitive information like passwords and credit card numbers. Adware tracks browsing habits to deliver targeted advertisements, often invading privacy and slowing down systems. Trojan spyware disguises itself as legitimate software while performing malicious activities in the background. System monitors comprehensively track all activities on a computer, including applications used, websites visited and files accessed, providing attackers with detailed insights into a user's behaviour.

Spyware can significantly impact users by stealing sensitive information, reducing system performance and causing financial or data loss. To protect against spyware, it is essential to use up-to-date antivirus and anti-spyware software, regularly update operating systems and applications, and exercise caution with downloads and email attachments.

Ransomware

Ransomware is a type of malware designed to block access to a computer system or encrypt files on the system until a sum of money (ransom) is paid. It poses a significant threat to individual users, businesses and even government agencies.

The technical underpinnings of ransomware involve several key components and techniques that enable its release, spread and execution. This typically involves an infection mechanism, such as phishing emails with malicious attachments or links. Once activated, ransomware typically initiates an encryption routine, targeting specific file types or entire storage systems. This ensures that files are securely locked and the decryption keys are nearly impossible to obtain without paying the ransom. After encrypting the victim's files, the ransomware displays a ransom note on the system, demanding payment in exchange for the decryption key. The note usually includes instructions on how to pay the ransom, often demanding payment in cryptocurrencies like Bitcoin to maintain the anonymity of the attackers.

THINK ABOUT APPLIED COMPUTING

7.12

Conduct internet research on the most recent ransomware attacks this year. How many users were affected? Are there any noticeable trends?

Weak passwords

A combination of a username and password is one of the most widely used security controls. Each user of an information system is allocated their own unique username, while, in most cases, a user is able to create their own password. Collectively, a username and password is often referred to as the login.

Some information systems require users to create a password that meets particular requirements. Commonly, for example, the password:

- must consist of at least eight characters
- must contain at least one upper-case letter
- must contain at least one lower-case letter
- must contain at least one symbol
- must not contain spaces
- cannot be ‘password’
- cannot be the same as the username.

Short or single-word passwords are a threat to security. These are vulnerable to brute-force attacks. Obvious passwords such as *password*, or users who use the same password for multiple information systems, are particularly susceptible.

A **brute-force attack** involves making repetitive attempts to access an information system using a variation of possible usernames and/or passwords. Often an attempt will be based on an established username – for example, an email address.

A bot can be developed that will attempt to guess the password that matches the username. On the first attempt the password input could be ‘a’, then ‘b’, then ‘c’, right through to ‘z’, then the bot will start over again but this time at ‘aa’, then ‘ab’, ‘ac’, etc.

TABLE 7.4 Password protection: the time it takes to discover a password

Password		Average time to discover ^b	
Number of characters	Possible combinations ^a	Human ^c	Computer ^d
1	36	3 minutes	0.000 018 second
2	1300	2 hours	0.000 65 second
3	47 000	3 days	0.02 second
4	1 000 000	3 months	1 second
5	60 000 000	10 years	30 seconds
10	3 700 000 000 000 000	580 million years	59 years

a Possible character combinations include the letters A–Z (capitals only) and numbers 0–9.
b Average time assumes the password would be discovered in approximately half the time it would take to try all possible combinations.
c Human discovery assumes one try every 10 seconds.
d Computer discovery assumes 1 million tries per second.

Outdated versions of software

A **software update**, sometimes referred to as a patch, is generally a small file that is downloaded and that fixes any bugs, improves any functionality issues, makes minor changes to the user interface or addresses any security issues with existing software applications installed on a device or network. Operating system updates are a common example of a software update.

These updates attempt to address any newly identified security threats that may make a digital system vulnerable to hackers or malware. Smartphone applications rely on regular software updates.

A **software upgrade** is a new version of the software. An upgrade generally involves a large number of significant changes to the application, including the addition of improved security and functionality and significant changes to the user interface. Upgrading from Windows 8.1 to Windows 10 is an example of a software upgrade.

One method used to distinguish a software update from a software upgrade is the use of a version number. Version numbers generally follow a convention whereby they are made up of three numbers in the form 'X.Y.Z'.

The first number (X) indicates the version number, and this changes when significant changes have been made to the software. The second number (Y) indicates a minor version upgrade where small changes have been made to functionality, while Z indicates an update addressing bugs or errors in the software. There are a wide range of version numbering conventions used.

Failing to update or upgrade software can lead to the following issues:

- vulnerability against newfound security risks
- missing out on new functionality of the software
- lower battery performance or slower processing times
- lower productivity
- decreased functionality related to the use of software that contains bugs or errors.

Preventative practices to reduce risks and mitigate threats

Preventative practices are intended to reduce the chance of data loss or theft due to security risks. Such practices may include software-based, process-based or hardware-based measures, each attempting to reduce the chance of different types of risks. Like all security measures, preventative practices aim to reduce the risk of data loss or theft, but they cannot absolutely guarantee security.

Software maintenance

Software maintenance involves making changes to software applications, after they have been installed, to fix faults, improve performance or add future functionality to the software. Software maintenance involves making changes to the programming code used to create the application.

Maintenance of software can be required for a variety of reasons. The changes made may be:

- *adaptive changes* – made to the software to adjust to changes in the operating environment
- *perfective changes* – made to the software to improve the functionality of the application
- *corrective changes* – made to fix bugs or errors in the application
- *preventative changes* – made to improve the reliability of the application or reduce the need for future maintenance.

Software updates and upgrades are examples of software maintenance. These files are made available to users via a download from the software developer.

Maintaining software helps to address any new security threats and improves the reliability of the software. This reduces the chance of data loss and improves functionality, allowing users to be more productive when using the application.

Operating system updates

An **operating system update** is a form of software update. Operating system updates are frequently released by organisations such as Apple, Google and Microsoft to improve the usability, functionality and security of an operating system.

When a new version of an operating system is released, users are quick to provide feedback about areas of the operating system that could be improved. As well, cybercriminals and hackers are quick to find vulnerabilities in operating systems, and the developers of the operating system thus constantly need to make changes to the code.

Operating system updates can also include updates to associated applications installed on the device.

Operating system updates aim to improve:

- protection against spyware, viruses and other malware
- protection against cyber attacks
- the overall performance of the device.

Software malware updates

As discussed on page 289, a **virus** is a software application that is written to cause some type of negative effect on a device or network. When a virus is executed, it normally duplicates itself in a range of locations on a device. A virus may be designed to damage, steal, modify or corrupt data.

System protection software contains antivirus functionality: the software will scan a device to detect whether any viruses have infiltrated the device. Antivirus software is used to scan a device or network in search of viruses. Antivirus software needs to be aware of the characteristics of a virus before it is able to identify a virus on a device or network. To do this, antivirus software uses virus definitions.

A **virus definition** is a collection of binary code that is used to identify a particular virus. The definition for each known virus is stored in a database attached to the antivirus software. As the software scans a device or network, it compares each file against the binary code stored in the virus definition database. If a virus is found, the antivirus software will either delete or quarantine the virus.

If the antivirus software is not aware that a virus exists, then it will not be able to detect that particular virus if it is present on the device or network. Since new viruses are being written every day, antivirus software needs to be constantly updated in order to be effective. This often requires the software to download updated virus definitions. For many antivirus software applications, this updating occurs automatically (often daily).

Application of firmware

Firmware is software that has been programmed onto a hardware device such as a television controller, hard drive, mouse or keyboard. The software is designed to give instructions to communicate with other devices in a system. It is semipermanent software: it stays the same unless it is updated. It cannot be uninstalled or removed. Firmware is subject to security threats and needs to be updated as required by the manufacturer.

Nmap is a free and open-source utility for network discovery and security auditing.



WebLink
Nmap

Metasploit is a widely used penetration testing framework.



WebLink
Metasploit

A back door is a method that bypasses normal authentication to allow unauthorised access to a system, often remaining hidden from standard security measures. It can be intentionally created for legitimate purposes or installed by malicious hackers.

A rootkit is a set of tools that enable an unauthorised user to gain and maintain control over a computer system while hiding their presence. Rootkits operate at a deep level within the operating system, making them difficult to detect and remove.

Staff procedures

Staff procedures are steps or processes that staff should follow in a given situation when dealing with data and information. Many organisations require staff to follow a series of procedures related to the use of data and information within the organisation, including procedures relating to:

- leaving devices unattended
- password requirements
- file naming conventions
- acceptable use of the system
- email protocols
- internet usage and downloading guidelines.

Training staff to follow these procedures when using data and information reduces the chance of data loss or threat.

Ethical hacking

Ethical hacking, also known as penetration testing or white-hat hacking, involves skilled professionals systematically probing and testing computer systems, networks or web applications to identify security vulnerabilities that could be exploited by malicious attackers. The primary goal is to strengthen the security infrastructure by proactively addressing potential weaknesses before they can be exploited.

The process begins with reconnaissance, where ethical hackers gather information about the target system. This phase includes both passive techniques, such as searching publicly available information, and active techniques, like scanning the network. Tools like Nmap (Network Mapper) are commonly used in this phase to scan for open ports, services running on the target system, and potential vulnerabilities.

Following reconnaissance, ethical hackers move to the scanning phase. Here, they use various tools to scan the target for vulnerabilities. Nessus and OpenVAS are popular tools for identifying known vulnerabilities and misconfigurations within the system.

Once vulnerabilities are identified, ethical hackers attempt to exploit them to gain access to the system. Metasploit, a widely used exploitation framework, allows ethical hackers to develop and execute exploit code against a remote target machine. This step is an important part of understanding how a malicious attacker could potentially infiltrate the system.

After gaining access, ethical hackers may try to maintain their presence in the system, simulating how a real attacker would establish long-term access. They might use techniques such as creating back doors or employing rootkits to stay hidden within the system.

Finally, ethical hackers may simulate the steps a malicious hacker would take to cover their tracks. This includes clearing logs and hiding files or network traffic to ensure their activities are not detected.

Ethical hacking provides several significant benefits. It allows organisations to proactively identify and fix security flaws, enhancing their overall security posture. It also helps in regulatory compliance, as many industries require regular penetration testing to meet various standards. Additionally, identifying vulnerabilities through ethical hacking aids in better risk management, allowing organisations to assess and manage risks more effectively.



vectorfusionart/Shutterstock

There are courses you can complete to become a certified ethical hacker, such as through EC-Council.



WebLink
EC-Council

FIGURE 7.25 Ethical hacking is a growing industry.

Key legislation

There are several key laws that affect how organisations ethically control the security and communication of data and information. These laws govern the collection and use of private information by both government and non-government organisations at both state and federal levels. Employers and government agencies have a legal and ethical responsibility to ensure that these laws are implemented within their organisations. Organisations must make employees and customers aware of their rights, as well as their responsibilities, in relation to these laws.

Privacy Act 1988

The *Privacy Act 1988* has been discussed in detail in Chapter 6. The 13 Australian Privacy Principles (APPs) are shown in Table 6.1 on page 207. The APPs that relate to how organisations ethically control the security and communication of data and information are:

- APP 1: Open and transparent management of personal information. Organisations must have a clearly articulated and accessible privacy policy. This policy should be freely available and easy for individuals to find. It should explain how and why the organisation collects, holds, uses and discloses personal information. The policy must also outline how an individual can access and correct their personal information held by the organisation.
- APP 6: Use or disclosure of personal information. Organisations are allowed to use or disclose personal information for the primary purpose it was collected. They may also use or disclose personal information for secondary purposes if the individual has consented or if it is otherwise permitted by law.
- APP 11: Security of personal information. Organisations are required to take reasonable steps to protect the personal information they hold from misuse, interference, loss, unauthorised access, modification or disclosure. This includes implementing security measures such as

encryption, access controls and secure storage to prevent unauthorised access and data breaches. Organisations must also have processes in place to detect and respond to data breaches, including notifying affected individuals and relevant authorities when required.

Privacy and Data Protection Act 2014

The *Privacy and Data Protection Act 2014* (Vic) was discussed in Chapter 6. The 10 Information Privacy Principles (IPPs) are shown in Table 6.2 on page 209. The IPPs that relate to how organisations ethically control the security and communication of data and information are:

- IPP 1: Personal information can only be used and disclosed for the primary purpose for which it was collected, or for a secondary purpose that would be reasonably expected.
- IPP 4: Information must be protected from misuse, loss, unauthorised access, modification or disclosure. Reasonable steps must be taken to destroy or de-identify personal information that is no longer needed.
- IPP 5: The organisation needs to be transparent about what it does with information.

Health Records Act 2001

The Victorian *Health Records Act 2001* was discussed in Chapter 6. The 11 Health Privacy Principles (HPPs) are summarised in Table 6.3 on page 211. The HPPs that relate to how organisations ethically control the security and communication of data and information are:

- HPP 2: Only use or disclose health information for the primary purpose for which it was collected or a directly related secondary purpose the person would reasonably expect. Otherwise, you generally need consent.
- HPP 4: Safeguard the health information you hold against misuse, loss, unauthorised access and modification. Only destroy or delete health information in accordance with criteria of HPP4.
- HPP 5: Document clearly expressed policies on your management of health information and make this statement available to anyone who asks for it.



RESEARCH

Patient privacy

Jordan is a medical student working as a receptionist at a local doctor's surgery during his mid-term break. His job is to greet patients as they arrive at the surgery, retrieve their medical records and alert the doctor of their arrival. At the end of their consultation, he collects the fee and makes another appointment if required.

The surgery database has detailed medical records on its 215 patients. Each record is stored under the patient's name and address. Jordan has downloaded a copy of the surgery's patient database onto his laptop as he wants to be able to refer to real-life examples when he is completing a group assignment for his university studies. He intends to share the medical records with the two other students in his group. Jordan's next rotation is at a hospital in regional Victoria and he intends to take his laptop with him.

- 1 Refer back to Table 6.2 on page 209. List the IPPs that are relevant to this case study, with a comment on how each is relevant.
- 2 Has Jordan breached any other Act? If yes, which one has he breached, and how?

Emerging trends in cyber security

Emerging trends in cyber security reflect the evolving landscape of technology, the increasing sophistication of cyber threats and the growing need for robust defence mechanisms. Five key areas that might have a significant influence are: security measures (multi-factor authentication), regulatory compliance, artificial intelligence-based (AI) threat detection, zero trust architecture and use of cryptography for protection strategies.

Security measures

Security measures such as multi-factor authentication (MFA) are gaining traction as standard security practices due to the inadequacy of simple password protection against increasingly sophisticated cyber attacks. MFA adds layers of security by requiring users to provide two or more verification factors to gain access to digital resources. This approach significantly reduces the risk of unauthorised access, as compromising multiple authentication factors is much more difficult for attackers. The adoption of MFA is becoming more widespread across various platforms and industries, enhancing security by protecting user identities and sensitive data.

Multi-factor authentication typically involves something the user knows (password), something the user has (security token or a smartphone app), and something the user is (biometric verification like fingerprints or facial recognition).

Regulatory compliance

As cyber threats grow, so do regulations and standards designed to protect sensitive information and ensure privacy. Compliance involves implementing and maintaining policies, procedures and security measures that meet these regulatory requirements, which can influence how data is collected, stored, used and shared. Non-compliance can result in significant financial penalties, legal repercussions and damage to reputation.

AI threat detection

By analysing vast data sets and identifying patterns, AI-based systems can detect anomalies that may indicate a cyber threat, often well before human analysts would. These systems learn and adapt over time, improving their ability to predict and mitigate potential attacks. AI-based threat detection allows for real-time security insights and proactive defence strategies.

These systems continuously monitor network traffic, user behaviour and system activities, analysing vast amounts of data to detect anomalies indicative of potential threats. By identifying these threats as they occur, AI allows for immediate alerts and responses, reducing the window of vulnerability.

Additionally, AI systems can predict potential attacks based on learned patterns and historical data, allowing organisations to implement defensive measures before an attack can occur, thereby enhancing overall security.

Some examples of AI-based threat detection systems include:

- Darktrace – uses machine learning to detect and respond to cyber threats in real-time by understanding and modelling network behaviour
- CrowdStrike Falcon – employs AI to provide endpoint protection, identifying and stopping breaches by analysing threat data across its customer base
- CylancePROTECT – uses AI to prevent malware infections by analysing the characteristics of files before they execute.

Quantum-resistant algorithms are cryptographic methods designed to secure communications against the potential future threat posed by quantum computers. Quantum computers are expected to have the capability to solve certain mathematical problems much more efficiently than classical computers. This capability includes breaking the cryptographic algorithms currently used for securing digital communications and data, such as RSA, which rely on the difficulty of factoring large numbers or solving discrete logarithm problems.

Zero trust architecture

Adopting a **zero trust** approach is increasingly vital as traditional network perimeters dissolve with the rise of remote work and cloud services. Unlike traditional security models that assume everything inside an organisation's network can be trusted, zero trust operates on the principle that trust should never be assumed, regardless of the source.

Zero trust means that just because someone or something is connected to a network, it does not automatically get access to everything on the network. Instead, every attempt to access a part of the network is checked to make sure it is allowed. This checking happens every single time, making sure that only the right people can access the right information or parts of the network, and they can only do so under the right conditions. It is a way of making networks more secure by never taking trust for granted, always verifying first.

Zero trust minimises the attack surface by requiring verification for every access request, regardless of its origin, helping to prevent unauthorised access and data breaches.

Cryptography for protection strategies

Cryptography secures information by transforming it into unreadable formats for unauthorised users, using encryption for data in transit and at rest. Quantum-resistant algorithms are also being developed in anticipation of quantum computing's potential to break traditional encryption methods. These algorithms are designed to be secure against both classical and quantum computing attacks, using mathematical problems that are believed to be difficult for quantum computers to solve.

Essential Eight

The **Essential Eight** is a set of strategies from the Australian Cyber Security Centre (ACSC) aimed at providing organisations with a baseline of cyber security measures that, when implemented effectively, can significantly reduce the risk of cyber threats. These strategies are designed to protect against various forms of cyber attacks, including malware infection, unauthorised information access, and system compromise.

It includes strategies such as application whitelisting, patching applications, configuring Microsoft Office macro settings, restricting administrative privileges, patching operating systems, multi-factor authentication and daily backup of important data.

Information Security Manual

The **Information Security Manual (ISM)** provided by the ACSC, offers guidelines to government and other organisations on protecting their information and systems. Specifically, for networking, it includes encryption, segmentation and segregation, network access controls, confidentiality and integrity of wireless network traffic, and wireless network footprint.

Encryption

The ISM emphasises the importance of encrypting data to protect its confidentiality and integrity, especially when transmitted over potentially insecure networks. The ISM provides guidelines on choosing strong encryption algorithms and managing encryption keys effectively to prevent unauthorised access to sensitive information.

Segmentation and segregation

Network segmentation and segregation involves dividing a network into smaller, manageable segments to control access and reduce the risk of lateral movement by malicious actors within the network. The ISM recommends implementing segmentation to isolate sensitive information and systems from the rest of the network. This can limit the damage from a cyber breach and make it harder for attackers to find and access valuable data.

Network access controls

Network access controls are mechanisms used to prevent unauthorised access to the network. The ISM guides organisations on implementing robust access controls that verify the identity and permissions of users and devices attempting to connect to the network. This includes the use of authentication methods, such as passwords and multi-factor authentication, and ensuring that users and devices have access only to the resources necessary for their roles.

Confidentiality and integrity of wireless network traffic

The ISM provides guidelines on securing wireless networks by ensuring the confidentiality and integrity of data transmitted over networks. This includes the use of strong encryption for wireless traffic, securing wireless access points and monitoring for unauthorised access attempts.

Wireless network footprint

The **wireless network footprint** refers to the physical coverage area of a wireless network. The ISM advises on minimising the wireless network footprint to reduce the risk of external attacks. This involves configuring wireless networks so that signals do not extend beyond the physical boundaries necessary for legitimate use, limiting the opportunity for attackers to access the network from outside the premises.

7

CHAPTER SUMMARY

Essential terms

AES (Advanced Encryption Standard) a symmetric encryption algorithm used for securing data

asymmetric digital subscriber line (ADSL) a digital line alternative for the small business or home user. ADSL transmits on existing standard copper telephone wiring. The ADSL2+ technology allows a faster transfer rate than the older ADSL rate – 20 Mbps compared with 8 Mbps

asymmetric encryption the use of public and private keys to decrypt messages

Bluetooth a protocol that uses short-range radio waves to transmit data along enabled devices, such as notebook computers, mobile telephones and printers

broadband router a basic router that connects a LAN to the internet, also functioning as a switch, a firewall and a wireless access point

brute-force attack a continual trial and error method to find out a password and username

Caesar cipher a substitution cipher that shifts characters by a fixed number of positions

cipher an algorithm for performing encryption or decryption

communications satellite a space station that receives microwave signals from Earth, amplifies the signals, and retransmits them back to Earth

cryptography the practice of securing information by transforming it into an unreadable (ecnrypted) format

data storage and backup systems software solution designed to store and protect digital data. These systems encompass various storage technologies, including hard disk drives (HDDs), solid-state drives (SSDs), network-attached storage (NAS), cloud storage and tape drives. Backup systems create duplicate copies of data to protect against data loss due to hardware failures, accidental deletion, cyber attacks or other disasters, ensuring data integrity, availability and continuity

denial-of-service (DoS) attack a malicious act that consists of flooding a network with traffic to make it unusable

downlink transmission from a communications satellite to a receiving Earth station

Earth-based station a communications facility that uses large, dish-shaped antennas to transmit and receive data from satellites

ECDH (Elliptic-Curve Diffie-Hellman) a key exchange protocol using elliptic-curve cryptography

ECDSA (Elliptic Curve Digital Signature Algorithm) a digital signature algorithm based on elliptic-curve cryptography

encryption the process of encoding or changing data so that an unauthorised user who reads the data would not be able to understand it

Essential Eight a set of eight cyber security strategies recommended by the Australian Cyber Security Centre

ethical hacking attempting to gain access to a device or network without authorisation, with the intention of informing the organisation of weaknesses with the current security strategy so changes can be made that improve overall security

fibre-optic cabling a wired transmission medium containing shards of glass that reflect pulses of light generated by small lasers or light-emitting diodes (LEDs)

firewall a network security device or software application that monitors and controls incoming and outgoing network traffic based on predetermined security rules. Firewalls establish a barrier between a trusted internal network (such as a company's intranet) and untrusted external networks (such as the internet), filtering network packets to prevent unauthorised access, malicious activities or data breaches

firmware software that has been programmed onto a hardware device

hacker a person who gains unauthorised access to a computer motivated by profit or a challenge, or in order to view restricted data

Information Security Manual (ISM) guidelines by the Australian Cyber Security Centre for securing information systems

internet a worldwide network in which billions of devices are connected together for the purposes of communication and data exchange

intranet a private network accessible only to authorised users within an organisation

intrusion detection system (IDS) a system, either hardware-based or software-based, that is used to monitor network traffic in an attempt to detect intrusions

intrusion prevention system (IPS) a system with the functionality of an IDS that also attempts to block or stop threats without the need for human intervention

key a series of bits (0s or 1s in binary code) that is used to apply an algorithm to a data packet to scramble (encrypt) and unscramble (decrypt) data that requires protection

Local Area Network (LAN) a network that is created when two or more devices are connected together in the same geographical area

malware short for 'malicious software'; programs designed to infiltrate and cause damage, disruption or access to a device or network without the user's knowledge or consent; includes viruses, worms, Trojan horses, adware, spyware, logic bombs and keyloggers

Mbps short for megabits per second; a unit of data transfer speed

microwave station Earth-based reflective dish that contains the antenna, transceivers and other equipment necessary for microwave communications

microwaves radio waves that can be used to provide high-speed transmission of both voice and data; data is transmitted through the air from one microwave station to another in a manner similar to the way radio signals are transmitted

mobile network also known as a cellular network; a network that uses telecommunications networks to allow users to communicate using their mobile devices

modem a device used to connect a device to a network using a copper cable

National Broadband Network (NBN) a fibre-optic backbone for the transmission of digital data covering a large part of Australia

near-field communication (NFC) digital technology using radio waves that allows contactless communication between devices over a very small distance, without the need to configure settings; examples include payWave and myki

network a collection of two or more digital system devices, connected by communications channels, that facilitates communications among users and allows users to share resources with one another

network access controls security measures to manage who can access a network

network diagram a visual representation of a network that shows the nodes that make up a network and how they interact

network security measures to safeguard computer networks from unauthorised access, data breaches, malware infections and other security threats

network segmentation and segregation dividing a network into segments to enhance security

node a network connection point, such as a desktop or mobile computer, a peripheral such as a printer or scanner, or a portable device such as a smartphone

operating system update a form of software update released to improve the usability, functionality and/or security of an operating system

optical fibre a strand of glass or plastic, as thin as a human hair, that uses light to transmit signals

Personal Area Network (PAN) a computer network for connecting an individual's devices within a limited range

polyalphabetic cipher a cipher that uses multiple substitution alphabets to encrypt data

private key a key that is only known by a user who has the key installed on their device

public key a key that is made available to other devices and is used to encrypt data packets that are being sent

router a device that is used to connect two separate networks together

RSA (Rivest-Shamir-Adleman) an asymmetric encryption algorithm widely used for secure data transmission

software maintenance the process of making changes to software applications, after they have been installed, to fix faults, improve performance or add future functionality to software

software update sometimes referred to as a patch; generally, a small file that is downloaded and that fixes any bugs in a piece of software

software upgrade replacing an old version of software with a new version

staff procedures steps or processes staff should follow in a given situation when dealing with data and information

standalone device any piece of computing equipment that can perform its function without the need for another device, computer or connection

switch a device that is used to connect multiple devices to a network

symmetric encryption use of one key to both encrypt and decrypt a message

threat anything that has the potential to cause harm to data and information stored and communicated between information systems

Transmission Control Protocol/Internet Protocol (TCP/IP) a network standard that manages the transmission of data by breaking it up into packets and transmitting the packets over the internet

twisted-pair cable cable made up of twisted-pair wires

twisted-pair wires two separate insulated copper wires that are twisted together to reduce interference

uplink transmission from an Earth-based station to a satellite

very high bitrate digital subscriber line (VDSL) a DSL technology offering higher data rates over copper telephone lines

Vigenère cipher a polyalphabetic cipher that uses a keyword to encrypt text

virtual private network (VPN) extends a private network across a public network and allows a user to use public networks such as the internet to send and receive data and information as if they were using a private network, such as an intranet

virus a computer program that can destroy files and alter the performance of the operating system; once a virus is in a computer, it can spread over a network to other connected computers

virus definition a collection of binary code that is used to identify a particular virus

Wide Area Network (WAN) a network that is situated in more than one geographical location

Wi-Fi communication a wireless transmission medium that distributes radio signals through the air over long distances

wired network a network in which the devices are connected using a physical cable

wireless access point a communications device that allows devices to wirelessly connect to an existing wired network

wireless broadband router a device that combines the functions of a basic router (connecting the LAN to the internet), a switch (for devices, such as a desktop computer, connected by cable), a firewall (security measure) and a wireless access point (to allow wireless connectivity)

wireless network a computer network that uses wireless data transfer between network devices

wireless network footprint the area covered by a wireless network's signal

Wireless Personal Area Network (WPAN) a type of personal network that uses wireless transmission media to transfer data between a user's personal devices

zero trust a security model that assumes no trust and verifies every access request

Important facts

- 1 A network is a collection of computers and devices connected by communication channels that facilitates communication among users.
- 2 Types of networks include Local Area Network, situated in one location, and Wide Area Network, in multiple locations.
- 3 A Personal Area Network is a network in which a user's individual devices are connected to each other.
- 4 Hardware required on a network includes a switch, to connect devices; a router, used to connect separate networks; and a modem, which allows data communication over a telephone line.
- 5 An intrusion detection system (IDS) is used to monitor network traffic in an attempt to detect intrusions. While an intrusion prevention system (IPS) contains the functionality of an IDS, it also attempts to block or stop threats, without the need for human intervention.
- 6 Communication methods used to transmit data include wired networks, wireless transmission – either Wi-Fi (802.11x) or Bluetooth – and mobile network standards (3G, 4G and 5G). Wired networks can use either twisted-pair or fibre-optic cabling.
- 7 An intranet is a private network accessible only to users within an organisation. The internet is a worldwide network, and a virtual private network (VPN) allows a user to send and receive data as if their computing devices were directly connected to the private network from a remote location.
- 8 A network diagram is used to visually represent a network.
- 9 Machine learning algorithms related to network security analyse vast amounts of network traffic data to identify patterns and behaviours typical of both normal operations and potential security threats.
- 10 Real-time monitoring and notification systems provide immediate alerts about potential security incidents.
- 11 Risks of using networks in a global environment include cyber security threats, data privacy threats, legal compliance issues and unauthorised network access.
- 12 Reducing risks and mitigating threats involves applying firmware updates, using multi-factor authentication, putting in place backup strategies and regularly updating software applications and operating systems with patches.
- 13 Ciphers are used to protect data and networks by providing security through the encryption of information. Three important ciphers in Applied Computing are the Caesar cipher, polyalphabetic cipher and Vigenère cipher.
- 14 Encryption uses keys. A key is used to scramble and unscramble data. A private key is a key that is only known by a user, and a public key is a key that is made available to other devices and is used to encrypt data packets. Important encryption methods used in Applied Computing are AES, RSA, ECDH and ECDSA.
- 15 A threat is anything that has the potential to cause harm to data and information stored and communicated between information systems.
- 16 A denial-of-service attack (DoS) is when hackers attempt to prevent legitimate users from accessing a service from an information system or network by flooding that system with authentication requests.
- 17 Malware is software that is designed to damage, disrupt or gain unauthorised access to an information system. It includes viruses, worms, spyware and ransomware.
- 18 Preventative practices to reduce risks include software maintenance, operating system updates, software malware updates, application of firmware and putting in place staff procedures to follow when dealing with data and information.
- 19 Ethical hacking involves attempting to gain access to a device or network without authorisation, with the intention of identifying weaknesses with the current security strategy, to inform the organisation about those weaknesses.
- 20 Key legislation and industry frameworks related to the handling of data and information include Essential Eight, the Information Security Manual (ISM), the *Privacy Act 1988* (Cwlth), the *Privacy and Data Protection Act 2014* (Vic) and the *Health Records Act 2001* (Vic).
- 21 Emerging trends in cyber security include five key areas: security measures, regulatory compliance, AI-based threat detection, zero trust architecture and use of cryptography.



TEST YOUR KNOWLEDGE

Networks

- 1 How does a WAN differ from a WPAN?
- 2 A school wants to connect all computers in a single building and enable students to connect their personal devices to teachers' workstations nearby. Which type(s) of networks would you use and why?
- 3 A company needs to connect its local office networks in different countries and allow employees to connect their smartphones to work computers nearby. Identify and describe the networks the company will need.
- 4 Describe the roles of routers and switches. How does each device contribute to network functionality and what are the key differences between their functions?
- 5 A company is experiencing network performance issues. Explain how a router and a switch could be used to optimise the network. Provide specific examples of the problems each device can solve.
- 6 How does an intrusion prevention system block threats to data and information?

Wired, wireless and mobile technologies

- 7 Compare the cost implications of deploying wired, wireless and mobile communication technologies in an organisation. Discuss how these costs can impact the decision-making process for choosing the appropriate technology.
- 8 A retail company is setting up a new branch with a main office and a storefront. The office staff will use laptops and the storefront will use tablets for transactions. The company needs a reliable and secure network connection between the office and storefront. Which type of communication technology is most appropriate: wired, wireless or mobile? Justify your choice considering cost, reliability and security.

Types of networks

- 9 Describe one similarity and one difference between an intranet and the internet.
- 10 Explain when a user would require the use of a virtual private network.
- 11 Explain the use of a network diagram.

Network risks and security threats

- 12 Explain how spyware operates to collect sensitive information from infected systems. What methods does it use to avoid detection and how can organisations protect against it?
- 13 Explain how multi-factor authentication (MFA) enhances the security of network systems. Provide an example of a situation where MFA could prevent unauthorised access even if a password is compromised.



- 14 Describe the key components of an effective backup strategy and how they help mitigate the impact of cyber attacks such as ransomware.
- 15 Provide an example of how outdated software can be exploited by cyber criminals and how updates can prevent such attacks.
- 16 Describe how a denial of service (DoS) attack can disrupt network services. What are the typical methods used in such attacks, and how can organisations protect their networks against them?
- 17 Discuss two risks associated with improper credential management.

Ciphers and encryption

- 18 Describe the process of encrypting and decrypting a message using the Vigenère cipher. Provide an example with a short plain text and a keyword to demonstrate how the Vigenère cipher enhances security compared to simple substitution ciphers.
- 19 Explain the difference between asymmetric and symmetric key encryption.

Ethical hacking

- 20 What is the purpose of ethical hacking, and how does it help organisations improve their security?
- 21 Name two tools that could be used by ethical hackers during the reconnaissance and scanning phases and briefly describe their functions.
- 22 What techniques do ethical hackers use to maintain their presence in a system after gaining access, and why is this step important?

Laws

- 23 Which law regulates the handling of personal data for all Australians?
- 24 Explain the Information Privacy Principles (IPPs) of use and disclosure.



APPLY YOUR KNOWLEDGE

Medical Answers is an organisation that provides administration and database support for small- to medium-sized medical clinics across Australia. Services offered by Medical Answers include:

- training for clinic staff in the use of database management systems (DBMS) software
- database maintenance and management
- 24-hour customer support.

At present, Medical Answers employs three staff and has a turnover of around \$2.5 million per year, though the business has been growing over the last couple of years and this growth is expected to continue over the next few years.

The computer office network at Medical Answers consists of three laptops, a printer, a file server, a switch, a router, a modem and a wireless access point.

The file server, router and wireless access point are all connected to the switch using twisted-pair cables. The three laptops then use wireless transmission to connect to the network via the wireless access point, while the router is connected to a modem, which is then connected to a telephone line used to connect the network to the internet.

To provide database maintenance and management, staff of Medical Answers require remote access to the network of each medical centre network.

Right now, the staff at Medical Answers do not use usernames and passwords on their network in the office. Neither the laptops, the Wi-Fi connection nor the file server require any authorisation.

The network has a firewall installed on the internet connection, and computer protection software is used to scan the network for malware, but this software has not been updated for more than 12 months.

When accessing a medical clinic's network remotely, all staff use a generic username and password. These are currently set as 'admin' and 'password' for each clinic, to make them easy to remember. Staff then download the database from a medical clinic to perform database maintenance, store a copy on the file server locally, then upload the updated database back to the medical clinic.

- 1 Describe the type of network used by Medical Answers.
- 2 Describe three threats to the data and information either communicated by or stored on Medical Answers' network. Evaluate the significance of each threat.
- 3 Discuss three legal or ethical issues that may arise from Medical Answers' current security practices.
- 4 Propose three security techniques that could be introduced to the network. Justify how each technique will improve security.

PREPARING FOR Unit 2

OUTCOME 2

Students are to respond to a teacher-provided case study to examine a cyber security incident or a network vulnerability, evaluate the threats to a network, and propose strategies to protect the security of data and information on the network.

Skills required

You will be required to demonstrate the following skills:

- describe the key components and capabilities of different networks
- examine a cyber security incident or a network vulnerability
- evaluate the impact of network vulnerabilities and threats to the security of data and information
- discuss possible legal and ethical issues arising from ineffective security strategies
- propose and justify strategies to protect the security of data and information within a network.

Documents required for assessment

You will be required to submit your written responses to the case study to your teacher.



EXAM PRACTICE

Unit 2 exam practice questions

For Unit 2, you need to ensure that you have a sound understanding of the required key knowledge as well as the ability to demonstrate the key skills. When revising, it is useful to relate the key knowledge to real-world applications to enhance comprehension. Real-world applications provide concrete examples of abstract concepts, making them more tangible and easier to understand. For example, describing the network of your school or home helps you demonstrate contextual understanding of the reasons we choose between wired, wireless and mobile networks.

You need to be familiar with how innovative solutions can help to solve modern problems, needs or opportunities. As the technological landscape we live in continues to grow, emerging technologies will be varied and potentially life-changing. Consideration of the impact of AI, how emerging technologies are affected by key legislations and regulatory frameworks, and the ethical issues surrounding the development of emerging technologies are all important elements of Outcome 1. Outcome 2 requires an understanding of the risks and threats to security that go alongside data transmission across the internet. You must practise evaluating existing or proposed systems and be able to provide strategies to protect data and information.

Thoroughly read and understand all questions, including the case study when provided. Underline command terms and key phrases in the question to help you understand the intention of the question, especially for theory and short-answer sections.

For multiple-choice questions, learn to use the process of elimination in questions where you are unsure of the answer. Often, you can narrow down the choices to two, increasing your chances of selecting the correct answer.

Short-answer questions require clear and concise responses. Focus on directly answering the question, avoiding unnecessary details. Demonstrate your knowledge by using appropriate technical terms. However, ensure that you fully understand the terms you use.

Practise managing your time effectively. Allocate time to each section based on its weight in the overall grade and your proficiency in that area. If time permits, review your answers, focusing on sections you found challenging. This can help catch and correct any mistakes or oversights.

Multiple-choice questions

Question 1 (1 mark) KK2.1.1

Which of the following best describes the role of curiosity in driving innovation towards achieving the Sustainable Development Goals?

- A. Curiosity fosters complacency and resistance to change.
- B. Curiosity encourages exploration, questioning and discovery of new solutions.
- C. Curiosity is irrelevant to innovation and achieving the SDGs.
- D. Curiosity limits creativity and problem-solving abilities.



Question 2 (1 mark) KK2.1.4

Which of the following characteristics are typically associated with creative and innovative solutions?

- A. Predictability and conformity
- B. Replication and imitation
- C. Originality and uniqueness
- D. Stability and rigidity

Question 3 (1 mark) KK2.1.9

When monitoring the progress of a software development solution, a manager might use _____ to keep informed about timeliness, costs and use of resources.

- A. annotated journals
- B. GitHub version control repository
- C. team software change logs
- D. Gantt charts

Question 4 (1 mark) KK2.2.3

The following data table compares features and characteristics of wired, wireless and mobile networks.

Feature/ characteristic	Wired networks	Wireless networks	Mobile networks
Infrastructure	Physical cables (e.g. Ethernet)	Radio signals (e.g. Wi-Fi)	Cellular towers
Speed	Typically high-speed	Speed varies based on signal strength and interference	Speed varies based on network coverage and technology (e.g. 4G, 5G)
Reliability	Generally more reliable due to stable physical connections	Susceptible to signal interference; may experience dropped connections	Moderate reliability; affected by signal strength and network congestion
Mobility	Not mobile; fixed locations	Offers mobility within signal range	Highly mobile; can maintain connectivity while moving between cell towers
Installation	Requires physical installation of cables	Easy to set up; minimal installation required	No installation required for end users; SIM card activation for devices
Security	Generally more secure due to physical connections	Susceptible to eavesdropping and signal interception	Security measures such as encryption and authentication are implemented
Coverage	Limited to the length of cables	Coverage area depends on signal strength and access points	Coverage varies based on network infrastructure and geographic location
Latency	Low latency due to direct connections	Latency can vary based on signal strength and network congestion	Moderate to high latency, depending on network load and technology
Cost	Higher initial setup costs for infrastructure	Lower initial set-up costs; ongoing subscription fees may apply	Variable costs based on usage and subscription plans



EXAM PRACTICE

Which network type is most susceptible to signal interference and coverage limitations?

- A. Wired networks
- B. Wireless networks
- C. Mobile networks
- D. Mesh networks

Question 5 (1 mark) KK2.2.7

In 2017 there was a malicious Wannacry attack that affected 230 000 Windows computers in 150 countries. The zero day vulnerabilities had been revealed a month after Microsoft had released a patch for the Windows OS vulnerabilities. The computers, once infected, could not access data on their network until released by the malicious actors who had distributed the program.

Wannacry is a type of:

- A. spyware.
- B. virus.
- C. worm.
- D. ransomware.

Short-answer questions

Sample case study: Smart Campus Initiative

A Victorian university has launched a Smart Campus Initiative, integrating emerging technologies to enhance student experience and efficiency. IoT devices monitored and optimised energy usage, AI-driven learning platforms provided personalised content, and blockchain secured academic records. AI-powered cameras and facial recognition systems improved security while respecting privacy. Augmented reality (AR) applications offered interactive campus tours and virtual labs.

Question 1 (4 marks) KK2.1.4

State and explain two characteristics of a creative and innovative solution. Use one of the solutions described in the case study as part of your explanation.

Question 2 (4 marks) KK2.1.6, KK2.1.8

Describe how empathising with the end user can assist when developing an innovative solution. As part of your description, refer to methods of data collection that could be used in order to achieve this.

Question 3 (2 marks) KK2.1.13

After gathering market research data, the university design team has chosen to prompt a generative AI with the specification requirements and design priorities. State two ethical issues that may arise from the use of artificial intelligence.



Question 4 (4 marks) KK2.1.10

Explain two testing and evaluation methods that could be implemented to evaluate the efficiency and effectiveness of the Smart Campus Initiative solutions.

Question 5 (4 marks) KK2.1.13

Propose two processes that the Smart Campus Initiative can put in place to ensure that AI-generated content is free from bias, discrimination and potential misuse for cyber attacks. Refer to any relevant legislation of frameworks in your response.

Question 6 (2 marks) KK2.2.9

Identify two potential threats to data and information in the Smart Campus Initiative system.

Question 7 (2 marks) KK2.2.1

How can advanced network security software like Darktrace AI protect against vulnerabilities in the Smart Campus Initiative?

Question 8 (2 marks) KK2.2.9

Explain two practices the university can put in place to mitigate security threats such as improper credential management and malicious software.

Question 9 (2 marks) KK2.2.12

The university employs an ethical hacker to help make its network more secure. What is the purpose of ethical hacking and how can it be used in the Smart Campus Initiative?

Question 10 (4 marks) KK2.2.13

How do key legislation and industry frameworks, such as the Essential Eight and the *Privacy Act 1988*, influence the ethical control of security and communication of data in the Smart Campus Initiative?

Index

- 32-bit computer systems 120
- 64-bit computer systems 120
- abstraction 145–6
- accessibility, of data 17
- accessibility aids 178–9
- accountability 213
- accuracy 12
 - of data 26
 - and fairness 26
- AES (Advanced Encryption Standard) 286
- alignment 83
- AlphaFold 190
- alternative execution 136
- alternative ideas, for innovative solutions
 - adapt 230
 - combine 230
 - compare 230
 - research 231
 - sleep on it 230–1
 - strip 230
 - substitute 230
 - visualisation 231
- American Psychological Association (APA) 11–12
- analysis and iteration, for usability testing 239
- analysis stage, of problem-solving methodology
 - functional requirements 224
 - non-functional requirements 224–5
 - scope of solution 225–6
 - solution constraints 225
 - solution requirements 223
- annotated diagrams 80–2
- Anscombe, Frank 71
- appearance 73
- appearance design tools 81–2
 - mock-ups 81–2
 - mock-ups vs. wire frame 82
- Apple Pay 184
- arguments/parameters 144–5
- arithmetic mean 46
- arithmetic operators 135
- array 122
- artificial intelligence (AI) 189–90
 - biased, discriminatory or harmful content 215
 - content and the potential for cyber-attacks 215
 - data generated by 9–10
 - and network security 280–1
 - problems with copyright 215
 - and privacy issues 33
 - threat detection 297
- assistive and wearable technologies 190
- assistive technology
 - accessibility aids 178–9
 - cognitive aids 180
 - home automation 180
 - mobility aids 180–1
 - personal emergency response systems 180
- asymmetric digital subscriber line (ADSL) 265
- asymmetric encryption 287
- augmented reality (AR) 193
- Australian Bureau of Statistics 15
- Australian Bureau of Statistics Act 1975* 9
- Australian Code for the Responsible Conduct of Research*, 2018 28
- Australian Privacy Principles 19–21
 - anonymity and pseudonymity 21
 - security of personal information 21
 - use/disclosure of personal information 21
- Australia's Artificial Intelligence (AI) Ethics Principles
 - accountability 213
 - contestability 213
 - fairness 213
 - human, social and environmental wellbeing 213
 - human-centred values 213
 - privacy protection and security 213
 - reliability and safety 213
 - transparency and explainability 213
- automation, emerging technologies 194–5
- backups 23
- backup strategies 282–3
- Best Practice Spreadsheet Modeling Standards (BPMS) 45
- bias 12–14
 - in graphic representations 13–14
 - small sample size 13
 - through sorting 13
 - timing 13
- biometric systems 214
- blockchain technology 185, 194
- Bluetooth 273
- Boolean data types 16–17, 121
- brainstorming 229
- broadband routers 265
- brute-force attack 291
- built-in functions 144
- Caesar cipher 283
- calculations 50–1
- cameras 188–9
- cell naming 41–2
- cell references 40–1
- census data 9
- chained selection 138
- character data type 15, 121
- character encoding 121
- chart 68
- chatbots 181
- ChatGPT 53, 189
- ciphers 283–5
- clarity, of data 17
- classes 147
- closed questions 7
- cognitive aids 180
- collecting data 221–2
- columns and data types 43
- communications satellites 275
- compilers 110, 153
- computerised wheelchair 181
- conditional operators 135
- confidentiality, of data 27
- constraints, of software solution
 - economic constraints 117
 - legal constraints 118–19
 - non-technical constraints 117–18
 - technical constraints 117
- contestability 213
- context, of data 18
- contrasts 87
- control structures
 - functions 143–5
 - iteration 140–3
 - selection 136–40
 - sequence 136
- conventions 85
- copyright 212
- Copyright Act 1968* 118, 212
- coxcomb chart 67
- creative and innovative solutions,
 - characteristics of 222–3
 - disruptiveness 223
 - effectiveness 222
 - originality 222
 - user-centred 223
- cryptocurrency 185
- cryptographic techniques, to protect networks

- ciphers 283–5
 - encryption 286–7
 - cryptography 298
 - curiosity and ingenuity 175–6
 - cyber attacks 25
 - cyberbullying, emerging technologies 195–6
 - cyber security, emerging trends in
 - AI threat detection 297
 - cryptography for protection strategies 298
 - Essential Eight 298
 - information security manual 298–9
 - regulatory compliance 297
 - security measures 297
 - zero trust architecture 298
 - cyber security threats 214, 281
-
- dashboards 66
 - data 3
 - Australian Privacy Principles 19–21
 - defined 5
 - ethical issues 25–33
 - and information 5
 - primary and secondary data 5–10
 - privacy 18–19
 - security to protect personal information 22–5
 - data and information
 - artificial intelligence, data generated by 9–10
 - census data 9
 - data collection methods 6–8
 - quality of 12–15
 - quantitative and qualitative data 8
 - sources 6
 - techniques and methods 6
 - database and spreadsheet software
 - cell naming 41–2
 - cell references 40–1
 - columns and data types 43
 - databases 43
 - foreign keys 43–4
 - macros, use of 45–6
 - primary keys 43
 - spreadsheet is not database 42
 - tables 43
 - databases 43, 86
 - data collection 6–8, 221–2
 - data extraction
 - database software 55–6
 - programming languages 56–7
 - spreadsheet software 54–5
 - DataGenix 216
 - data integrity 14
 - data privacy 281
 - data storage and backup systems 267
 - data structures, types of
 - array 122
 - list 122
 - record and field 123
 - data types 15, 119
 - Boolean 121
 - defined 15
 - numeric 119–21
 - software tools 15–18
 - text 121
 - data visualisations 87–101
 - categories of 63–5
 - charts 89–90
 - defined 61
 - lines and arrows 88–9
 - purposes of 62–5
 - software 90–101
 - text styles 87–8
 - titles 87
 - types of *see* types of, data visualisations
 - date/time 120–1
 - debugger 110
 - debugging 152
 - decrypt 22
 - defending networks 267–8
 - de-identify 18
 - demographic data 9
 - denial-of-service (DoS) attack 288
 - dependencies 218
 - design ideas 229–32
 - design tools
 - appearance design tools 81–2
 - functionality design tools 82–3
 - using temporary guidelines 83–4
 - deskilling, economic issues of emerging technologies 196–7
 - Dexcom G7 190
 - differential backup 23
 - digital subscriber line 266
 - digital systems, functions and capabilities
 - assistive technologies 178–81
 - financial services 181–5
 - global positioning system (GPS) 185–7
 - robotics 187
 - traffic management 188–9
 - dilemma 26
 - disruptiveness 223
 - divide-by-zero errors 153
 - documenting and monitoring progress 217–21
 - Gantt chart 217–20
 - project journals 220
 - project table 217
 - version control 221
 - DO/WHILE loop 141
 - downlink 275
 - drones 191–2
-
- Earth-based station 275
 - ECDH (Elliptic-Curve Diffie-Hellman) 287
 - ECDSA (Elliptic Curve Digital Signature Algorithm) 287
 - economic constraints 79
 - economic issues, emerging technologies
 - deskilling 196–7
 - job losses 197–8
 - editor 110
 - educational software development project 227–8
 - effectiveness 80, 222, 247
 - efficiency 83, 247
 - electronic funds transfers (EFT) 184–5
 - electronic validation 58
 - Email verification 181
 - emerging technologies 189
 - artificial intelligence, machine learning, neural networks 189–90
 - assistive and wearable technologies/Internet of Things (IoT) 190
 - automation 194–5
 - blockchain 194
 - cyberbullying 195–6
 - drones, microcontrollers, nanosatellites and robotic devices 191–2
 - economic issues 196–8
 - games development/multimedia programming 192
 - human interactions 199
 - interpersonal skills 199
 - legislation and *see* legislation and emerging technologies
 - mixed realities 193
 - smart devices in health care 191
 - emerging trends, in cyber security
 - AI threat detection 297
 - cryptography for protection strategies 298
 - Essential Eight 298
 - information security manual 298–9
 - regulatory compliance 297
 - security measures 297
 - zero trust architecture 298
 - empathy mapping 226–8
 - encapsulation 146–7
 - encryption 22, 286–7
 - Energy Superpowers 92
 - Essential Eight 298
 - ethical hacking 294
 - ethical issues 214
 - consent forms 28–9
 - defined 25
 - inaccurate/incomplete data 26–7
 - ownership and control of data 27–8
 - participation information statements 30–2
 - personal data and information, misuse of 32–3
 - repurposing and sharing of data by AI systems 33
 - transparency 26
 - ethical responsibility 27
 - Eurostat 15
 - event 147

- Excel 4, 91
 - existence checking 151
 - expected results 152
 - exploratory visualisations 63–5
 - exploring multiple perspectives, in
 - ideation 228
 - exponent 16
 - extracting data 54–6
 -
 - fairness 26, 213
 - fibre-optic 266
 - fibre-optic cable 270–1
 - field 123
 - filenaming 85
 - filtering 58
 - financial services 181–5
 - firewall 24, 267
 - firmware 293
 - firmware updates 282
 - floating point 120
 - floating point number 16, 119
 - flowcharts 125–6
 - flow of execution 158
 - footnotes 11
 - foreign keys 43–4
 - FOR loop 141–2
 - formats, defined 85
 - formats and conventions
 - databases 86
 - data visualisations 87–101
 - filenaming 85
 - spreadsheets 86–7
 - Form Controls 91
 - fraud detection 182
 - full backup 23
 - functions 143–5
 - functionality design tools 82–3
 - IPO charts 82
 - query designs 83
 - functional requirements 224
 - function declaration 144
 - function definition 144
 - function visibility 144
 -
 - games development/multimedia
 - programming 192
 - Gantt chart 217–20
 - global positioning system (GPS) 185–7
 - global variables 144
 - Google Forms 4
 - Google Pay 184
 - Google Public Data Explorer 15
 - graphical user interface (GUI) 111
 - group interview 6
 -
 - hacker 269
 - harm, of data 27
 - header comment 149
 - Health Records Act 2001* 210–12, 296
 - high-level programming language 109
 - home automation 180
 - home networks 262, 263
 - human, social and environmental
 - wellbeing 213
 - human-centred values 213
 - human interactions, emerging
 - technologies 199
 - hypotheses 3
 -
 - ideation
 - alternative ideas 230–1
 - brainstorming 229
 - for exploring multiple perspectives 228
 - mind mapping 229
 - improper credential management 288–9
 - incremental backup 23
 - infinite loop 141
 - infographics, characteristics of 65–6
 - information 5
 - for communication and
 - decision-making 12
 - and data *see* data and information
 - Information Security Manual (ISM) 298
 - confidentiality and integrity of wireless
 - network traffic 299
 - encryption 298
 - network access controls 299
 - segmentation and segregation 299
 - wireless network footprint 299
- informed consent 28
- ingenuity 175–6
- innovative solution 217
 - characteristics of creative and 222–3
 - collecting data 221–2
 - curiosity and ingenuity 175–6
 - design ideas 229–32
 - documenting and monitoring progress
 - 217–21
 - emerging technologies 189–99
 - empathy mapping 226–8
 - evaluating solution 248–50
 - evaluation criteria 247–8
 - expert review 239–40
 - for exploring multiple
 - perspectives 228
 - functions and capabilities of digital
 - systems 178–89
 - legislation and emerging technologies
 - 206–16
 - preferred design 233–8
 - solution specifications 223–8
 - testing 240–4
 - testing the solution 244–6
 - United Nations’ Sustainable
 - Development Goals 176–7
 - user testing 238–9
- integer 16, 119
- integer overflow 120
- integrated development environment
 - (IDE) 111, 160
- integrity 14
- intellectual property rights 118
- interactive charts 91–101
- interface 111
- internal documentation 149–51
- internet 276–9
 - IP addresses 277–8
 - TCP/IP 278–9
 - virtual private network (VPN) 279
- Internet of Things (IoT) 189, 190
- interoperability 212
- interpersonal skills, emerging
 - technologies 199
- interpreter 110, 153
- interviews 7, 10
- intranets 276–7
- intrusion detection system (IDS) 268
- intrusion prevention system (IPS) 268
- intrusions 268
- IP addresses 277–8
- IPO chart 82
- iteration 140–3
-
- job displacement 214
- job losses, economic issues of emerging
 - technologies 197–8
-
- Kestrel’s empathy map 228
-
- large language models (LLMs) 9
- legal compliance 281
- legislation and emerging technologies
 - artificial intelligence (AI), issues arising
 - from 214–16
 - Australia’s Artificial Intelligence (AI)
 - Ethics Principles 212–13
 - biometric systems collecting and storing
 - data 214
 - Copyright Act 1968* 212
 - cyber security threats 214
 - and ethical issues 214
 - Health Records Act 2001* 210–12
 - job displacement 214
 - Privacy Act 1988* 206–8
 - Privacy and Data Protection Act 2014*
 - 208–10
- library study room booking system 157
- Lightship Augmented Reality Developer
 - Kit (ARDK) 193
- list 122
- Local Area Network (LAN) 261–2
- local variables 144
- location-based factors 182
- logical operators 135

- logic error 152
- low-code development 112
- low-level programming language 109
-
- machine learning (ML) 189–90, 280–1
- macros 45–6
- maintainability 117
- malicious software 289–90
- malware 25, 289
- mantissa 16
- manual validation 58
- Mathematica 57
- Mbps 267
- median 46
- memory leak 153
- method 147
- microcontrollers 191–2
- Microsoft Entra 194
- Microsoft Excel 75
- Microsoft Forms 4
- microwaves 273, 274
- microwave station t 273
- mind mapping 229
- Minitab 4
- misinformation, of data 27
- misuse, of emerging technologies 198
- mixed realities 193
- mobile network 275
- mobility aids 180–1
- mock-ups 81–2
 - vs. wire frame 82
- mode 46
- modem 261
- multi-factor authentication (MFA) 181, 282, 297
 - defined 181
 - email verification 181
 - location-based factors 182
 - one-time passcode 182
 - personal identification number (PIN) 181
 - QR code verification 181–2
 - secret questions 182
 - security token 181
 - SMS-based verification 181
- MusicPlayer class 145–6
-
- naming convention
 - camel case 148
 - Hungarian notation 148–9
 - snake case 148
- nanosatellites 191–2
- National Broadband Network (NBN) 271
- National Privacy Principles 19
- natural language processing (NLP) 9
- near-field communication (NFC) 184, 273–4
- need/opportunity 217
-
- nested selection statement 140
- network access controls 299
- network diagrams 279
- networking hardware and software
 - broadband routers 265
 - digital subscriber line 266
 - fibre-optic 266
 - router 264–5
 - switch 266–7
- networks, advantages 261
- networks, types
 - Local Area Network 261–2
 - Wide Area Network (WAN) 262–3
 - Wireless Personal Area Networks (WPANs) 263
- network security 267
 - artificial intelligence and network security 280–1
 - cryptographic techniques to protect networks 283–7
 - defending networks 267–8
 - emerging trends in cyber security 297–9
 - ethical hacking 294–5
 - global environment 281–2
 - key legislation 295–6
 - key network hardware and software components 260–4
 - network diagrams 279–80
 - networking hardware and software 264–7
 - preventative practices 292–4
 - reducing risks and mitigating threats 282–3
 - security threats on 287–92
 - technical underpinnings of networks 276–9
 - wired, wireless and mobile communications technology 268–76
- network segmentation and segregation 299
- neural networks 189–90
- Nielsen’s usability heuristics 240
- node 261
- non-functional requirements 224–5
 - maintainability 117
 - portability 116
 - reliability 116
 - robustness 116
 - usability 116
- non-technical constraints 80
- ‘normal’ curve 46–7
- NVivo 4
-
- object 111
- objective 248
- object-oriented programming
 - abstraction 145–6
 - classes 147
 - encapsulation 146–7
 - methods 147–8
- observation 10
- one-time passcode 182
- OpenAI 53
- open-ended questions 7–8
- operating system updates 293
- optical fibre 270
- originality 222
- outdated versions of software 291–2
- ownership and control of data 27–8
-
- parameters 144
- participation information statements 30–2
- pass by reference 144
- pass by value 144
- passwords 24
- patches 153
- payment terminal 184
- Personal Area Network (PAN) 263
- personal emergency response systems 180
- personal identification number (PIN) 181
- personal information 21
- Philips HealthSuite 191
- phishing 25
- pie chart 69–71
- plagiarism 11
- planning, usability testing 239
- platform 110
- point-to-point camera systems 188
- polyalphabetic cipher 283–5
- portability 116
- preferred design 233–8
 - methods for developing 237–8
 - pseudocode 233–4
 - site maps 234
 - storyboards 234–6
- preventative practices, reduce risks and mitigate threats
 - application of firmware 293
 - operating system updates 293
 - software maintenance 292–3
 - software malware updates 293
 - staff procedures 294
- primary data 5–10
- primary keys 43
- primary sources 6
- privacy 18–19
 - of data 27
- Privacy Act 1988* 19, 206–8, 295–6
- Privacy Amendment (Enhancing Privacy Protection) Act 2012* 19
- Privacy and Data Protection Act 2014* 19, 208–10, 296
- privacy protection and security 213
- private key 286
- problem-solving methodology 223
- procedural programming languages 143

- procedure 143
 - product 237
 - programming editor 110
 - programming language, processing
 - features of
 - arithmetic, logical and conditional operators 134–5
 - control structures 136–43
 - functions 143–5
 - internal documentation 149–51
 - object-oriented programming 145–8
 - programming naming conventions 148–9
 - testing and debugging techniques 152–60
 - validation techniques 151–2
 - variables 134
 - programming languages and AI
 - low-code development 112
 - readability and maintainability improvements 113
 - project 217
 - project journals 220
 - project table 217
 - proof of concept 237
 - properties 111
 - prototyping 237–9
 - pseudocode 124–5, 233–4
 - pseudonym 21
 - public key 287
 - Python 56–7, 98
-
- QR code verification 181–2
 - qualitative data 4, 8, 69
 - advantages and disadvantages of 4–5
 - defined 4
 - quality of, data and information
 - accuracy 12
 - freedom from bias 12–14
 - integrity 14
 - relevance 15
 - reliability 15
 - Qualtrics 4
 - quantitative data 3, 8
 - advantages and disadvantages of 4–5
 - defined 3–4
 - quantum-resistant algorithms 298
 - queries 48, 49
 - query designs
 - artificial intelligence, usage of 53–4
 - JOINS 51–2
-
- R 57, 98
 - R1C1 notation 40
 - range checking 151
 - RC notation 40
 - real-time monitoring and notification 281
 - record and field 123
 - recruitment, usability testing 239
 - red-light cameras 188
 - reducing risks and mitigating threats, to
 - networks
 - backup strategies 282–3
 - firmware updates 282
 - multi-factor authentication (MFA) 282
 - software and malware updates 283
 - referencing 10
 - referencing primary sources
 - examples of 11–12
 - interviews 10
 - observation 10
 - surveys 10
 - regulatory compliance 297
 - relational database management system (RDBMS) 51
 - relevance 15, 17
 - relevant 15
 - reliability 15, 116
 - and safety 213
 - reliable 15
 - REPEAT/UNTIL loops 143
 - representing designs
 - mock-ups 123–4
 - pseudocode 124–5
 - research, types of 3–4
 - research questions 3
 - robotic devices 191–2
 - robotics 187
 - robustness 116
 - rose chart 67
 - router 264–5
 - RSA (Rivest-Shamir-Adleman) 287
 - runtime error 152
-
- satellites 185
 - scope of solution 225–6
 - secondary data 5–10
 - secondary sources 6
 - secret questions 182
 - security, to protect personal information
 - backups 23–4
 - encryption 22–3
 - firewall 24–5
 - malware protection 25
 - physical security controls 22
 - usernames and passwords 24
 - security measures 297
 - security threats, on networks
 - denial-of-service attacks 288
 - improper credential management 288–9
 - malicious software 289–90
 - outdated versions of software 291–2
 - weak passwords 291
 - security token 181
 - selection 136–40
 - sequence 136
 - site maps 234
 - size, of data 17
 - small sample size (bias) 13
 - smart devices in health care 191
 - Smithsonian Institution 15
 - SMS-based verification 181
 - software and malware updates 283
 - software development tools 110–11
 - software functions
 - statistics for analysis 46–7
 - statistics for MS Excel 47–8
 - software maintenance 292–3
 - software malware updates 293
 - software solution
 - characteristics of 113
 - constraints 117–19
 - functional requirements 114–15
 - non-functional requirements 116–17
 - scope, determining 113–14
 - software tools, for data types
 - Boolean 16
 - character data type 15
 - data and information, characteristics of 17–18
 - date/time 16
 - floating point 16
 - integer 16
 - string 16
 - software update 291
 - software upgrade 292
 - solution 217
 - solution constraints 225
 - solution requirements 223
 - solution specifications 223–8
 - source code 110
 - sources 6
 - referencing primary sources *see*
 - referencing primary sources
 - speed camera 188
 - spoofing email 25
 - spreadsheets 86–7
 - Spreadsheet Standards Review Board (SSRB) 45
 - staff procedures 294
 - standalone device 260
 - standard deviation (SD) 46–7
 - static 61
 - Statistical Package for the Social Sciences (SPSS) 4
 - storyboards 234–6
 - string 121
 - string of characters 16
 - structure, of data 17
 - Structured Query Language (SQL)
 - calculations 50–1
 - extracting data 54–6
 - Mathematica 57
 - Python 56–7
 - queries 49
 - query designs 51–4
 - validation of data 57–61

- subjective 248
- subroutine 143
- sum 46
- SurveyMonkey 4
- surveys 7, 10
- sustainability 177
- switch 265–7
- symmetric encryption 286
- syntax 110
- Tableau 75, 91
- tables 43
- task delegation 221
- TCP/IP 278–9
- technical constraints 79
- technical underpinnings, of networks
 - internet 277–9
 - intranets 276–7
- techniques and methods 6
- test cases 152
- testing, usability testing 239
- testing and debugging techniques
 - boundary values 155–8
 - breakpoints 153–4
 - debugging statements 154
 - test data and test tables 154–5
 - trace tables 158–60
 - types of errors 152–63
- testing and evaluating innovative solutions
 - evaluating the solution 248–50
 - evaluation criteria 247–8
 - expert review 239–40
 - testing 240–4
 - testing the solution 244–6
 - user testing 238–9
- testing table 244
- theories 3
- timing (bias) 13
- trace tables 152
- traffic management 188–9
- Transmission Control Protocol/Internet Protocol (TCP/IP) 276
- transparency and explainability 213
- truth table 137
- twisted-pair cable 269–70
- twisted-pair wire 270
- type checking 151
- types of, data visualisations
 - components of 66–73
 - constraints 79–80
 - dashboards 66
 - functional requirements 77
 - infographics 65–6
 - non-functional requirements 77–9
 - scope of solution 80
- unauthorised network access 281
- UN Data 15
- unencrypted data 22
- United Nations' Sustainable Development Goals 176–7
- Unity 192
- usability 77, 116
- usability testing 239
- user-centred 223
- usernames 24
- users' personal information 25
- user testing
 - prototyping 238–9
 - usability testing 239
- validation 58, 241
 - databases 59–61
 - spreadsheets 58–9
- validation techniques
 - existence checking 151
 - range checking 151
 - type checking 151
- variables 14
 - accessing 134
 - initialisation of 134
 - storing data in 134
- version control system 150, 221
- very high bitrate digital subscriber line (VDSL) 265
- vested interest, in research outcome 12–13
- Vigenère cipher 283
- virtual assistants 181
- virtual currencies 184–5
- virtual private network (VPN) 279
- virtual reality (VR) 193
- virus 293
- virus definition 293
- weak passwords 291
- WeChat Pay 184
- WHILE loop 140–1
- Wide Area Network (WAN) 261–3
- Wi-Fi communications 272–3
- wired communication technology
 - fibre-optic cable 270–1
 - National Broadband Network 271
 - twisted-pair cable 269–70
- wire frame 82
- wireless access point 265
- wireless broadband routers 265
- wireless communication technology
 - Bluetooth 273
 - communications satellites 275
 - microwaves 274
 - near-field communication 273–4
 - Wi-Fi communications 272–3
- wireless network 263
- wireless network footprint 299
- Wireless Personal Area Network (WPAN) 263
- Wolfram Mathematica 75
- World Health Organization (WHO) 15
- zero trust architecture 298
- Zipline 192

Flexible online learning designed to support you

Nelson MindTap puts **you** at the centre

Access resources and content that make learning simpler yet smarter to help you go beyond drill and practice.



Develop your understanding through the wide range of weblinks and digital resources.

Revise using quizzes to practice your skills and build your confidence.

Navigate your own path, accessing the content and support you need whenever you need it in your learning journey.

Find everything you need to access your Nelson MindTap course at
cengage.com.au/nelsonmindtap