Computing Technology Stage 5 (Year 9 or 10) – teacher support resource

**Software development – developing apps and web software**

# Teacher support resource

**Teacher note:** this resource has been designed to facilitate conversion into a student booklet by removing the answers within the response windows. Teacher notes can be deleted before distributing to students.

Student name:

Class:

Teacher:

Contents

[Teacher support resource 1](#_Toc180668911)

[Unit overview 4](#_Toc180668912)

[Assessment task 1 overview 5](#_Toc180668913)

[Submission details 7](#_Toc180668914)

[Steps to success 8](#_Toc180668915)

[What is the teacher looking for? 10](#_Toc180668916)

[Assessment task 2 overview 11](#_Toc180668917)

[Steps to success 12](#_Toc180668918)

[What is the teacher looking for? 13](#_Toc180668919)

[Glossary 14](#_Toc180668920)

[The design and production process 17](#_Toc180668921)

[Identifying and defining 18](#_Toc180668922)

[Apps and web-based tools 18](#_Toc180668923)

[Explore inputs, storage, transmission, processes and outputs in apps or web-based tools 22](#_Toc180668924)

[Functional and non-functional requirements 27](#_Toc180668925)

[The social impacts, and ethical and legal responsibilities in apps and web-software 29](#_Toc180668926)

[Researching and planning 40](#_Toc180668927)

[Understanding challenges in software development 41](#_Toc180668928)

[Object-oriented programming 43](#_Toc180668929)

[Elements and features of code 46](#_Toc180668930)

[The perspective of diverse groups 52](#_Toc180668931)

[Exploring the impact of app and web software on challenges 55](#_Toc180668932)

[Design principles and issues relevant to apps 58](#_Toc180668933)

[Self-assessment 68](#_Toc180668934)

[Evaluate whether solutions meet specific requirements 69](#_Toc180668935)

[Data privacy and cybersecurity in software development 71](#_Toc180668936)

[Data protection and cybersecurity 79](#_Toc180668937)

[Representing data 83](#_Toc180668938)

[Data types 84](#_Toc180668939)

[Producing and implementing 89](#_Toc180668940)

[Flowcharts 89](#_Toc180668941)

[Pseudocode 91](#_Toc180668942)

[Desk checking 95](#_Toc180668943)

[Real-world problems that can be solved by an app 100](#_Toc180668944)

[Error type in programming 102](#_Toc180668945)

[Producing and implementing 105](#_Toc180668946)

[Record of project development 106](#_Toc180668947)

[Practical development 108](#_Toc180668948)

[Testing and evaluating 110](#_Toc180668949)

[Explore careers in software development 111](#_Toc180668950)

[References 114](#_Toc180668951)

# Unit overview

In this unit, students will develop a fundamental understanding of developing apps and web software.

During Weeks 1 to 8 of the learning sequence students will gain an understanding of the computational, design and systems thinking used in developing an app or web software. A range of apps and web software will be investigated that allow students to understand how innovation, enterprise and automation have inspired the evolution of computing technology.

During Weeks 9 to 18 of the learning sequence, students will design and test a system, creating an app which is coded and iterative in design. To develop their coding skills, students work to design, produce and evaluate algorithms and implement them in an object-oriented programming language. Students manage, document and explain individual work practices.

During Weeks 19 to 20 of the learning sequence, students showcase their project to the class and seek self- and peer-review. Students also investigate careers in app and web software development industries.

# Assessment task 1 overview

**Type of task**: working in pairs or small groups, research, justify and develop a 3-minute pitch for an app product.

**Outcomes being assessed**:

A student:

* selects and applies safe, secure and responsible practices in the ethical use of data and computing technology **CT5-SAF-01**
* applies iterative processes to define problems and plan, design, develop and evaluate computing solutions **CT5-DPM-01**
* manages, documents and explains individual and collaborative work practices **CT5-COL-01**
* communicates ideas, processes and solutions using appropriate media **CT5-COM-01**

[Computing Technology 7–10 Syllabus](https://curriculum.nsw.edu.au/syllabuses/computing-technology-7-10-2022) © NSW Education Standards Authority (NESA) for and on behalf of the Crown in the State of New South Wales, 2022.

**Suggested weighting: 20%**

Work collaboratively to research, justify and pitch a mobile phone app to solve an identified problem or need.

Investigate a real-world problem or need that can be satisfied by developing a mobile application (app).

The task will include using tools to define the needs of the app, brainstorming potential ideas, seeking feedback and creating a wireframe considering inputs, storage, transmission, processes and outputs.

There are several steps involved in this task. Read each point below carefully and make sure you read the ‘Steps to success’ and the marking criteria as this will provide you with specific detail of what you are required to complete to be successful in this task.

You are required to:

1. Identify a real-world problem that can be satisfied using a mobile app.
2. Brainstorm a variety of options or ideas that would satisfy this real-world need.
3. Conduct a peer-review of your ideas.
4. Describe the input, output, transmission and storage for your chosen web app in relation to the software being used.
5. Identify functional and non-functional requirements of the app.
6. Design a wireframe of your app.
7. Develop an engaging and informative **pitch** to justify your wireframe. Create a pitch deck that clearly outlines the following:
8. The real-world problem you are addressing.
9. Your brainstormed ideas – describe how your ideas came about and how they apply to the development of your app.
10. The feedback from your peers – what did your peers tell you about your idea and how did you take their feedback into consideration?
11. Identify your final idea, explaining why you decided to go with this app.
12. Outline the input, output, transmission and storage for your app, along with the functional and non-functional requirements.
13. Show your wireframe, including the feedback you received from your peers.

**What is a pitch?**

A pitch is the act of presenting your app idea to an audience (in this case, your classmates and teacher). Your pitch needs to be persuasive because you want your audience to see that your solution to your real-world problem is beneficial and will be useful to a wider audience.

A good pitch should:

* engage your audience
* take the audience on a clear and logical journey
* leave the audience wanting to use your app.

Some suggestions when creating your pitch:

* Your pitch deck should be engaging, but not overwhelming. Avoid slides with too much text or information on them, for example, numerous animations or busy transitions.
* Use a font that is easy to read. The font type and colour should be accessible to all people. You can read more about [accessibility](https://accessibe.com/blog/knowledgebase/accessible-presentation-decks) when designing for further information.

**What does it mean to be persuasive or write in a persuasive way?**

When an author is writing in a persuasive manner, their aim is to present a viewpoint to the reader using evidence and facts to accept their argument. In this task, you will write in a persuasive manner to convince potential buyers or investors to use your app solution.

**Links to support:**

* [What is a Pitch Deck?](https://www.pitchvest.com.au/post/what-is-a-pitch-deck)
* [The secret to successfully pitching an idea (4:46)](https://www.youtube.com/watch?v=l0hVIH3EnlQ).

## Submission details

Students can submit their work digitally and complete a 3-minute presentation or video to pitch their app idea.

# Steps to success

|  |  |
| --- | --- |
| Steps | What I need to do |
| Outline a real-world problem or need that can be satisfied using a mobile app | * Research a variety of possible problems and needs. * Outline the chosen real-world problem or need that can be satisfied using a mobile app * Identify the needs of the user |
| Brainstorm a variety of ideas for the chosen real-world problem or need | * Brainstorm multiple ideas * Consider the following when creating ideas * Does it meet the problem or need? * How easy is it to create or produce this web app? * What are the time requirements to produce? * Does your group know how to make this on the designated web app software? If not, what learning will have to be done to be able to create it? |
| Peer feedback on brainstormed ideas | * Seek peer-feedback on all ideas * From the feedback, choose one final idea that will be the focus |
| Input, output, transmission, processes and storage | * In a table, describe the input, output, transmission, processes and storage for the chosen web app in relation to the software being used and the problem or need |
| Identify functional and non-functional requirements | * Specify both the functional and non-functional requirements of your app, including stating the purpose of a system, describing user cases and developing test cases of inputs and expected outputs * For example: functional requirements which may include user security such as authentication, verification emails to users, usability, user requirements and business requirements * Non-functional requirements may include speed of subscription or purchase cost, user motivation or engagement |
| Wireframe | * Create a wireframe of your chosen idea, including * what it will look like and what input, output and processing will occur * mock graphical user interface (UI) designs, colour, images and screen elements. * Conduct a peer-review (feedback) for your wireframe |
| Pitch | * Develop an engaging and informative pitch deck communicating all sections of the task including the final idea and wireframe that is no more than 3 minutes in length * Peer-feedback should be given on each wireframe * Work collaboratively |

# What is the teacher looking for?

The teacher is looking for a thorough understanding and application of the app development process, starting from identifying real-world problems or needs that can be addressed through a mobile app to creating a comprehensive wireframe and pitch.

The teacher is looking for creative ideas for the chosen problem or need and the application of peer-feedback to improve the final idea that is communicated well within the pitch.

# Assessment task 2 overview

**Type of task**: working in pairs or small groups, develop a mobile phone app product.

**Outcomes being assessed**:

A student:

* selects and applies safe, secure and responsible practices in the ethical use of data and computing technology **CT5-SAF-01**
* applies iterative processes to define problems and plan, design, develop and evaluate computing solutions **CT5-DPM-01**
* manages, documents and explains individual and collaborative work practices **CT5-COL-01**
* communicates ideas, processes and solutions using appropriate media **CT5-COM-01**
* designs, produces and evaluates algorithms and implements them in a general-purpose and/or object-oriented programming language **CT5-OPL-01**
* designs and creates user interfaces and the user experience **CT5-DES-01**

[Computing Technology 7–10 Syllabus](https://curriculum.nsw.edu.au/syllabuses/computing-technology-7-10-2022) © NSW Education Standards Authority (NESA) for and on behalf of the Crown in the State of New South Wales, 2022.

**Suggested weighting: 30%**

Work collaboratively to create your idea for a mobile phone app and solve your identified problem or need.

Collaboratively develop a working version of the wireframe presented in Assessment task 1 to solve the identified problem or need.

This task will include using tools to create a prototype that has full functionality. Identify errors and correct them and seek additional peer-feedback to evaluate the final design.

# Steps to success

|  |  |
| --- | --- |
| Steps | What I need to do |
| Name and logo | * Create a name and logo for the prototype that is reflective of the chosen problem or need |
| General-purpose programming | * Utilise appropriate general-purpose programming to create the working prototype. This includes correct handling of data being added, updated and deleted and conforming to the constructs of the language |
| Screen elements | * Choose appropriate screen elements that are appropriate for the role * Utilise all of the following: buttons, text boxes, drop-down menus and multiple screens |
| Prototype functionality and user experience | * Within the prototype, use a variety of aspects from the programming language or software that increase the functionality of the app * Within the prototype, use Intuitive navigation and smooth interactions. |
| Testing and feedback | * Conduct rigorous testing of the prototype with the team * Utilise peer-feedback to test the prototype, incorporating feedback to enhance design and functionality |
| Collaborative development | * Collaboratively develop the prototype * Work with the team through all stages of the task |

# What is the teacher looking for?

The teacher is looking for a prototype that meets the identified need, utilises appropriate general-purpose programming and has intuitive navigation for improved user experience.

The teacher is looking for collaborative development and communication while creating the app or web software product and responding to feedback throughout the whole process.

# Glossary

Many of the following words will gather more meaning to you as you work through this booklet.

Each time you see a new word in bold throughout this workbook you can add its definition in the table below in case you need to refer back to it later.

|  |  |
| --- | --- |
| Word | Definition |
| Algorithm | An algorithm is a procedure used for solving or performing a computation. |
| App | A software program designed for a specific purpose to run on mobile devices or on a personal computer. An abbreviation of the word ‘application’. |
| Cyber safety | A term that refers to behaviour and the precautions that may be exercised when providing personal information in an online or digital environment. |
| Cybersecurity | Cybersecurity is the protection of internet-connected systems such as hardware, software and data from cyberthreats. |
| Data | A discrete representation of information using number codes. Data may include characters (for example, alphabetic letters, numbers and symbols), images, sounds and/or instructions that, when represented by number codes, can be manipulated, stored and communicated by digital systems. For example, characters may be represented using ASCII (American Standard Code for Information Interchange) code or images may be represented by a bitmap of numbers representing each ‘dot’ or pixel. |
| Desk checking | Desk checking is the process of manually reviewing the source code of a program. It involves reading through the functions within the code and manually testing them, often with multiple input values. |
| Event-driven programming | Event-driven programming is a programming paradigm in which the flow of the program is determined by events such as user actions (mouse clicks, key presses), sensor outputs or message passing from other programs or threads. |
| Flowchart | A graphical representation of the sequence of operations in an information system or program. Different symbols are used to draw each type of flowchart. |
| Functional requirement | Functional requirements define the software's goals, meaning that the software will not work if these requirements are not met. |
| General-purpose programming language | A coding or programming language used to write computer software. It uses letters, numbers and symbols arranged in a prescribed format (language) to instruct a computer how to carry out specific tasks. Also known as text-based programming. |
| Logical operators | Logical operators are generally used for combining 2 or more relational statements. An operator used to compare logical expressions that returns a result of true or false. Common logical operators include AND, OR and NOT. |
| Non-functional requirement | A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. They are contrasted with functional requirements that define specific behaviour or functions. |
| Object-oriented programming (OOP) language | A type of programming language that organises code around data, or objects, rather than functions and logic. An object is a data field that has unique attributes and behaviour. |
| Prototype | A trial product or model built to test an idea or process to inform further design development. Its purpose is to see if and how well the design works and is tested by users and systems analysts. A prototype can be both a physical object and exist in digital form. |
| Pseudocode | Writing in plain English line by line (step by step) what you want the computer program to do. |
| Relational operators | A relational operator is a programming language construct or operator that tests or defines some kind of relation between 2 entities. |
| Test case | A test case is a set of actions performed on a system to determine if it satisfies software requirements and functions correctly. |
| Use case | A use case is a description of the ways in which a user interacts with a system or product. |

**Teacher note:** for students with an English as an additional language or dialect (EAL/D) background, the glossary can be provided complete so that they have additional time to understand the key terms using bilingual dictionaries. The glossary can be provided to students in their preferred communication mode.

# The design and production process

Throughout your study of Computing Technology, you will learn about design processes and how to apply them. You will explore different types of design processes and learn how to apply them in your design project.

The design and production process:

* involves a sequence of organised steps which provide a solution to design needs and opportunities
* may take a few seconds or minutes, such as when you select what clothes to wear, or may take years as in the case with the design of a motor vehicle
* may involve one person or may involve many people
* may be simple or complex, depending on the task
* involves questioning (or evaluating) throughout the iterative process.

Figure 1 – flowchart of design and production process

Design and production process diagram.

A flowchart labelled 'Ongoing evaluation' with a 2-headed arrow indicating both directions. 
The steps of the flowchart are as follows:
1. Identifying and defining: identify and define the needs, opportunities and wants of a computing challenge, practise the technical skills, develop evaluation criteria. 
2. Researching and planning: research, generate and practise ideas, be creative and propose new approaches to problems, explore new design opportunities. 
3. Producing and implementing: build and implement ideas, apply a variety of skills and techniques to create products that meet set criteria, modify and iterate solutions. 
4. Testing and evaluating: test and evaluate solutions/products, evaluate quality and effectiveness against the criteria, make judgements throughout the solution and use these to refine the product.
After testing and evaluating is a big arrow called 'Review if required to improve' and it goes all the way back up to the first part of the flowchart, indicating a cycle.

# Identifying and defining

## Apps and web-based tools

 Teacher introduces the concept of apps and web-based tools. As a class watch [What is a Web App? Web App vs. Native App (1:48)](https://www.youtube.com/watch?v=qt6gSW-uYKI).

**Activity 1**: define the following terms in the space below.

1. What is an app?

|  |
| --- |
| **Sample answer:**  A program that is designed to perform a specific function directly for the user. Examples include Instagram and Uber. |

1. What is a web software?

|  |
| --- |
| **Sample answer:**  Software that runs on your web browser. Examples include Google workspace (G Suite) and Canva. |

**Activity 2:** How have apps and web software evolved to meeting the changing needs of society?

[Brainstorm](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/542?clearCache=3d32c820-442f-7d68-d3ca-b2add7a710e8) ideas on how apps and web software have evolved to meet the changing needs of society.

Students should think about how society has changed over time and how these changes have influenced the development of apps and web software.

Students can also think about specific industries to connect real-world examples.

|  |
| --- |
| **Sample discussion points:**  **Evolving technology**  As society becomes more reliant on technology, the demand for apps and websites that cater to various needs continues to grow. From communication and entertainment, to shopping and education, technology plays a crucial role in our daily lives.  **Mobile revolution**  The widespread adoption of smartphones and tablets has transformed the way we access information and engage with digital content. This shift has led to a surge in mobile app development to ensure that websites and services are optimised for smaller screens and touch interactions.  **User-centric design**  The increasing importance of user experience has pushed developers to prioritise the usability and accessibility of apps and websites. Today, creating intuitive interfaces and providing seamless navigation have become fundamental aspects of development.  **Personalisation and customisation**  People now expect apps and websites to provide personalised experiences that cater to their individual preferences. Developers are leveraging data analytics and artificial intelligence to deliver targeted content, recommendations and suggestions to users.  **Connectivity and social integration**  Social media has revolutionised how we connect and communicate with others. Apps and websites now integrate social features to allow users to share content, collaborate and interact with others in real-time.  **Changing work dynamics**  The needs of the modern workforce have also influenced app and web development. Remote work and the gig economy have given rise to productivity apps, project management tools and platforms for freelancers, enhancing productivity and connectivity.  **Sustainability and social impact**  Society's increasing focus on sustainability and ethical practices has influenced app and web development too. Developers are now creating solutions that promote eco-friendly practices, social responsibility and inclusivity. |

**Activity 3:** research task

Choose a specific industry, such as:

|  |  |
| --- | --- |
| * health care | * construction |
| * finance | * retail |
| * education | * food and beverage |
| * travel | * transportation |
| * fitness and wellness |  |

In pairs or small teams, research and discuss how apps and web software have impacted the industry you have chosen and how the industry has evolved in response to societal needs and opportunities.

Consider how these tools have improved efficiency, accessibility or user experience**.**

Using Canva, create a one-page infographic to communicate your findings.

|  |
| --- |
| **Sample research on industries:**   * Healthcare: improved patient care and streamlined operations. * Finance: digital banking, investment and trading platforms. * Education: remote learning platforms, personalised learning experiences such as Khan Academy or Duolingo. * Travel and tourism: booking and reservation systems, destination guides and trip-planning tools. * Fitness and wellness: workout tracking and coaching apps, health and wellness monitoring devices. * Construction: project management and collaboration tools. * Agriculture: precision agriculture technologies, marketplace and supply chain management platforms. * Retail: e-commerce platforms, in-store technology integration for example immersive shopping experiences or digital signage. * Food and beverage: food delivery platforms, digital menu and ordering systems. * Transportation: ridesharing and mobility services, public transit and navigation apps. |

**Activity 4:** research the history of apps and web-software.

Research the history and evolution of apps and web-software.

Start by reading about the [history of mobile apps](https://inventionland.com/blog/the-history-of-mobile-apps/) and the [evolution of mobile apps](https://arkenea.com/blog/evolution-of-mobile-apps/) and take note of key dates for significant events and changes throughout time.

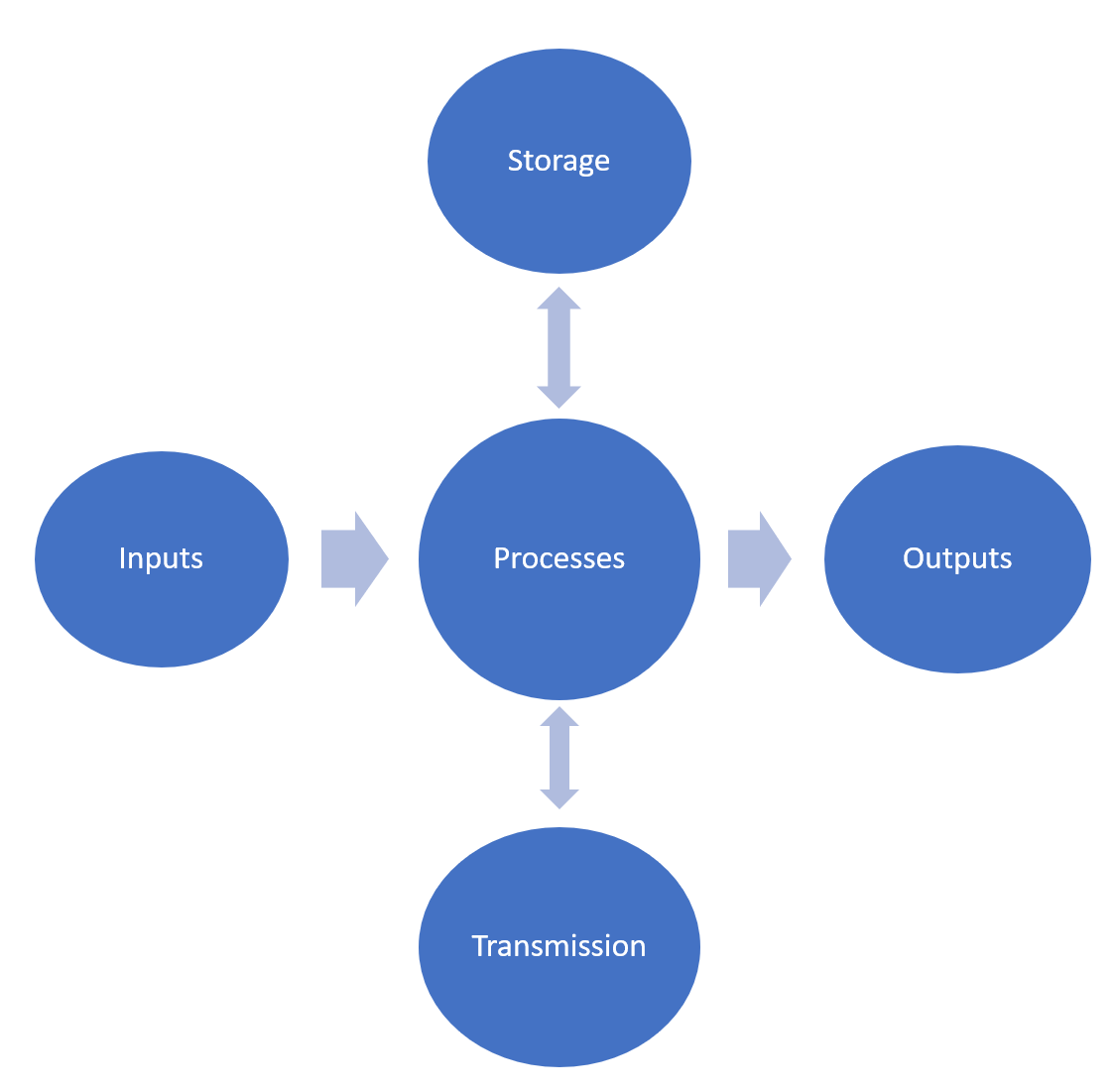
**Teacher note:** this activity can be completed individually or in pairs. Students can demonstrate their use of web-based software by creating their timeline through software such as Canva or Adobe Express.

## Explore inputs, storage, transmission, processes and outputs in apps or web-based tools

The concept of inputs, storage, transmission, processes and outputs in apps and web software refers to the fundamental elements that govern how these interactive experiences function.

* **Inputs** represent user actions or external data.
* **Storage** maintains the state and history of the system.
* **Transmission** deals with data exchange.
* **Processes** include computations and interactions within the system.
* **Outputs** manifest as visible or audible feedback to users.

Figure 2 – system components



**Activity 5:** exploring inputs, processes and outputs

****As a class watch [CS Basics: Input Process Output](https://www.youtube.com/watch?v=CBf-jIn44X0) (2:50).

After watching the video, complete a [Think, Pair, Share](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/645) activity.

* **Think** about how you would define the following in your own words: input, processes and output.
* **Pair** with the person next to you, how do your definitions differ?
* **Share** and report your definitions to the class.

|  |
| --- |
| **Report your findings below:** |

**Activity 6:** group brainstorming activity

****In small groups, [brainstorm](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/542?clearCache=3d32c820-442f-7d68-d3ca-b2add7a710e8) 5 apps and web-based software that the group uses regularly. For each example, outline the following:

* input
* processes
* transmission
* storage
* output.

|  |  |
| --- | --- |
| App and web-based software | Outline of input, processes, transmission, storage and output |
| Google Docs | **Sample answers:**  **Input:** text input, file uploads and collaborative editing.  **Processes:** text editing, collaborative tools for example comments, suggested edits and revision history.  **Transmission:** real-time collaboration and sharing options.  **Storage:** cloud storage and version history.  **Output:** completed document, collaboration insights and export options. |
| Snapchat | **Sample answers:**  **Input:** multimedia input for example, photos, videos, direct messages, filters and lenses.  **Processes:** multimedia editing, story creation and AR.  **Transmission**: real-time messaging, stories and curated content.  **Storage:** temporary storage and memories.  **Output:** shared content, story views and discover content. |
| Kahoot | **Sample answers:**  **Input:** assignment creation and student submission.  **Processes:** quiz hosting, game mechanics and feedback.  **Transmission:** device synchronisation, response submission and results displaying.  **Storage:** quiz data storage and account data.  **Output:** quiz results, engagement data and learning insights. |
| Spotify | **Sample answers:**  **Input:** music selection, personalisation, user interactions for example like/dislike songs and adding songs to a playlist.  **Processes:** music streaming, recommendation algorithms and playlist creation.  **Transmission:** streaming protocol for example transmission of audio, data synchronisation and offline mode.  **Storage:** music library, user data and offline cache.  **Output:** music playback, personalised recommendations and social sharing. |

**Activity 7:** inputs, storage, transmission, processes and outputs in apps or web-based tools

Choose an app or web-based tool within the following categories:

* health and fitness tracker (Fitbit)
* video conferencing (Zoom)
* online shopping (Amazon).

****Describe the input, process, transmission, storage and output for this web-based software.

|  |
| --- |
| **Sample answer:**  Health and fitness tracker (for example Fitbit)   * Input: user's physical activity, heart rate, sleep patterns and dietary information. * Storage: user profiles, activity history and health-related data. * Transmission: wireless data synchronisation between the tracker device and the companion app. * Processes: data analysis, goal tracking, activity recognition algorithms and personalised recommendations. * Output: activity summaries, sleep reports, heart rate monitoring and achievement badges. |

**Teacher note:** this activity could be an extension to the group task where teams are allocated a different real-world example and they create a diagrammatical representation which links all the components together to further consolidate student understanding.

## Functional and non-functional requirements

Functional requirements are features or functions that enable users to accomplish their tasks (user requirements).

Non-functional requirements are how the system should perform (user expectation).

As a class read about [functional and non-functional requirements](https://www.altexsoft.com/blog/business/functional-and-non-functional-requirements-specification-and-types/#:~:text=Functional%20requirements%20define%20what%20a,also%20known%20as%20quality%20attributes.) and watch [What are Non-functional Requirements and How Do They Work?](https://www.youtube.com/watch?v=fc-5HJPBZMQ&t=40s) (9:28).

**Activity 8:** defining key concepts

Research and define the following terms in relation to developing apps and web software.

1. What is a functional requirement?

|  |
| --- |
| **Sample answer:**  The features and capabilities that the app must have to solve the problem or meet the need.  For example, the website will have a homepage. |

1. What is a non-functional requirement?

|  |
| --- |
| **Sample answer:**  The quality attributes of the app, such as security, performance and user experience. For example, Level 4 encryption. |

1. What are use cases?

|  |
| --- |
| **Sample answer:**  Use cases are descriptions of how users will interact with the app to achieve specific goals or tasks. They outline the steps a user will take and the expected results. A use case for a social media app might describe how a user creates a new post, including steps like logging in, composing the post, adding media and publishing it. |

1. What are test cases?

|  |
| --- |
| **Sample answer:**  Test cases are specific scenarios and conditions that are used to verify that the app behaves as expected. Test cases are designed to validate that the app functions correctly under various circumstances, such as different input values or user actions. A test case for the social media app might involve checking that a new post appears in the user's feed after it's published. |

**Activity 9:** functional and non-functional requirements

Working in small groups, consider the following scenarios in Table 1 and identify the functional and non-functional requirements of the scenario.

Table 1 – functional and non-functional requirements

|  |  |  |
| --- | --- | --- |
| Scenario | Functional requirements | Non-functional requirements |
| Online banking app | * Allow users to view account balances * Allow users to transfer funds between accounts | * Security: the system must have a secure login and encryption to protect user data |
| E-commerce website | * Allow users to add items to a shopping cart * Allow users to make payments for purchases | * Performance: the website should load quickly and handle many concurrent users |
| To-do list app | * Users should be able to create an account with a unique username and password * Users can create tasks with details such as title, description, due date, priority and category | * Reliability: the app should always be reliable and available for use |
| Online learning platform | * Students and teachers can create accounts with personal information and login credentials * Teachers can create and publish courses with lectures, assignments, quizzes and resources | * Performance: the platform should load quickly and handle concurrent user interactions without slowdowns |

## The social impacts, and ethical and legal responsibilities in apps and web-software

When developing apps and web-software, developers must consider the social impact and ethical and legal responsibilities surrounding their work.

**Activity 10:** social, ethical and legal responsibilities

****Working in small groups, read and discuss the social, ethical and legal responsibilities in Table 2 below. Consider each side of the dilemmas listed and outline how developers may navigate these issues.

Table 2 – social, ethical and legal responsibilities

|  |  |  |
| --- | --- | --- |
| Issue | Definition | Explanation |
| Social impact | The effects that apps and web software can have on individuals and society, including how they influence behaviour, relationships and communities. | Games can shape how people interact, communicate and collaborate.  They can promote positive behaviours like teamwork and problem-solving, but they can also lead to issues like addiction and social isolation. |
| Ethical responsibility | Making moral choices and decisions while developing or engaging with apps and web software. | Software developers have a responsibility to create apps and web-based software that respect diverse cultures, avoid harmful stereotypes and promote fairness and inclusivity.  Users also have ethical responsibilities, such as treating others with respect with the software. |
| Legal responsibility | Rules and regulations set by governments or authorities that must be followed during software development, distribution and use. | Software must adhere to laws related to age restrictions, content ratings, copyright and data protection.  Breaking these laws can lead to legal consequences for both developers and players. |

After you have read and discussed Table 2 above, outline how social impact and ethical and legal responsibilities relate to apps and web software.

Think about the positive and negative for each, how the app and web software should be and what the developer will need to consider.

Group your ideas together in Table 3 below, then answer the question in full sentences in the space provided.

Table 3 – positive and negative

|  |  |  |  |
| --- | --- | --- | --- |
| Effect | Social impact | Ethical responsibility | Legal responsibility |
| Positive |  |  |  |
| Negative |  |  |  |

|  |
| --- |
| **Write your response here:** |

**Activity 11:** social and ethical scenarios

****Working in small groups, students read and discuss the social and ethical scenarios. Students are required to look at both sides of the scenario and provide how the developers would navigate this dilemma.

**Social and ethical scenario 1: data privacy versus personalisation**

In the development of apps and web-based software, developers often face the dilemma of balancing data privacy concerns with the desire to personalise user experiences.

On one hand, personalised recommendations and tailored content enhance user engagement and satisfaction. However, to achieve this personalisation, developers must collect and analyse vast amounts of user data, raising concerns about privacy infringement and potential misuse of personal information.

|  |
| --- |
| **Possible discussion points:**  Developers must navigate this dilemma by implementing privacy-enhancing technologies, obtaining informed consent from users and adopting transparent data practices. |

**Social and ethical dilemma 2: algorithmic bias versus fairness**

Software developers must consider the challenge of addressing algorithmic bias and ensuring fairness in automated decision-making systems. Apps and web-based software often rely on algorithms to make predictions, recommendations and decisions that impact users' lives. However, these algorithms may inadvertently perpetuate biases based on factors such as race, gender or socioeconomic status, leading to unfair outcomes and discrimination.

|  |
| --- |
| **Possible discussion points:**  Developers must grapple with the ethical implications of algorithmic bias and strive to moderate biases through rigorous testing, diverse data representation and algorithmic transparency. |

**Activity 12:** ethical dilemma group activity

****In groups, students are to read the scenarios relating to social and ethical issues and legal responsibilities. Consider the dilemma and answer the guiding questions as a team. Once completed report back to the class.

**Scenario 1: privacy concerns in social media platforms**

A popular social media platform is under scrutiny for its data privacy practices. It has been accused of collecting and sharing user data without adequate consent.

Table 4 – ethical dilemmas and guiding questions for Scenario 1

|  |  |
| --- | --- |
| Ethical dilemmas | Guiding questions |
| * How can the platform balance the need for informed consent regarding data collection with the desire to provide personalised services and content? * How would the platform address concern about data ownership and giving its users control over their personal information? | 1. How can the platform ensure transparency and informed consent regarding data collection and sharing practices? 2. What measures should be in place to protect user data from unauthorised access and misuse? 3. How should the platform address user concerns about data privacy while maintaining personalised services? |

**Scenario 2: legal dispute**

Two software development companies are involved in a legal dispute over intellectual property rights. One company accuses the other of copying proprietary code and design elements.

Table 5 – ethical dilemmas and guiding questions for Scenario 2

|  |  |
| --- | --- |
| Ethical dilemmas | Guiding questions |
| * How should software developers navigate the boundaries of intellectual property protection? * What legal actions can be taken to resolve the intellectual property dispute? | 1. What steps should software developers take to ensure they are not infringing on the intellectual property of others? 2. How can software companies encourage innovation and creativity while respecting copyright and patent laws? 3. What ethical considerations arise when developing software that builds upon existing technologies or ideas? |

**Activity 13:** ethical considerations, potential risks and strategies with copyright and IP laws

****Intellectual property refers to creations of the mind. It could include a brand, logo, invention, design or artistic work. Intellectual property rights protect these ideas and give the creator rights to profit from their work. Explore the [Australian Government – IP Australia](https://www.ipaustralia.gov.au/understanding-ip) website for more information.

Copyright provides legal protection for people who express original ideas and information in certain forms. The most common forms are writing, music and moving images. Copyright does not protect ideas or information, only the original expression of ideas or information. Explore the [Copyright basics – The Attorney General's Department](https://www.ag.gov.au/rights-and-protections/copyright/copyright-basics#:~:text=Copyright%20provides%20legal%20protection%20for,expression%20of%20ideas%20or%20information.) website for more information.

Reverse engineeringis the process of analysing and extracting information from an app’s code, data and behaviour and can help a developer understand how an app works, find security vulnerabilities, change or modify features or create a new app based on an existing one. Read more about the [reverse engineering of mobile apps](https://www.linkedin.com/advice/3/how-do-you-analyze-document-results-reverse#:~:text=Reverse%20engineering%20mobile%20apps%20is,apps%20based%20on%20existing%20ones.).

****In your groups, discuss the ethical considerations, potential risks and strategies that could be employed to comply with copyright and IP laws.

Table 6 – ethical considerations, potential risks and strategies

|  |  |
| --- | --- |
| Scenario | Ethical considerations, potential risks and strategies |
| A software developer uses code from a publicly available GitHub repository without checking the license terms. | **Ethical considerations:**   1. Respect for intellectual property – ethical developers should respect the ownership and rights of the original code creator. 2. Transparency – developers should be transparent about the sources of their code and adhere to license terms. 3. Avoiding plagiarism – using code without proper attribution or permission can constitute plagiarism and is ethically unacceptable.   **Potential risks:**   1. Legal liability – using code without checking license terms can lead to legal consequences, such as copyright infringement claims. 2. Reputation damage – being associated with plagiarism or IP violations can harm the developer's reputation within the software development community. 3. Loss of trust – users and collaborators may lose trust in developers who do not adhere to ethical and legal standards.   **Strategies for ensuring compliance:**   1. License verification – always check the license terms of code repositories before using code from them. 2. Adherence to license terms – respect the conditions set by the license, such as providing attribution or sharing modifications. 3. Use of permissive licenses – preference code with permissive licenses that allow reuse with fewer restrictions. 4. Seek permission – if unsure about license terms, seek permission from the code owner before using their code. |
| A company copies and modifies software from a competitor without obtaining permission. | **Ethical considerations:**   1. Fair competition – ethical considerations include fair competition and respecting the intellectual property of competitors. 2. Honesty and integrity – copying and modifying software without permission is dishonest and lacks integrity. 3. Legal compliance – ethical developers should comply with copyright and IP laws and avoid infringing on others' rights.   **Potential risks:**   1. Copyright infringement – copying and modifying software without permission can lead to allegations of copyright infringement. 2. Legal consequences – the company may face legal actions, fines, or injunctions for IP violations. 3. Reputational damage – being known for unethical business practices can damage the company's reputation and trustworthiness.   **Strategies for ensuring compliance:**   1. Conduct due diligence – before using or modifying software, ensure it is not protected by copyright or other IP rights. 2. Obtain permission – seek permission from the software owner or licence holder before copying or modifying their software. 3. Use open-source alternatives – if possible, use open-source software or create original solutions instead of copying competitors. 4. Legal review – consult legal experts to ensure compliance with copyright and IP laws. |
| A small game developer includes copyrighted artwork in their game without purchasing the appropriate licences. | **Ethical considerations:**   1. Respect for creative work – ethical considerations include respecting the creative work of artists and creators. 2. Fair compensation – artists should be compensated fairly for the use of their artwork. 3. Transparency – developers should be transparent about the sources of artwork used in their games.   **Potential risks:**   1. Copyright infringement – using copyrighted artwork without proper licences can lead to copyright infringement claims. 2. Legal consequences – the developer may face legal actions, penalties, or compensation from the copyright owner. 3. Reputational damage – being associated with copyright infringement can harm the developer's reputation and credibility.   **Strategies for ensuring compliance:**   1. Purchase licences – always purchase the appropriate licences or obtain permission from the copyright owner before using copyrighted artwork. 2. Use royalty-free resources – use royalty-free or licensed artwork that allows commercial use without infringing on copyrights (creative commons). 3. Create original artwork – whenever possible, create original artwork or collaborate with artists to produce unique content for the game. 4. Copyright notices – include copyright notices and attributions for all copyrighted artwork used in the game. |
| A programmer reverse-engineers a popular software application to create a competing product. | **Ethical considerations:**   1. Fair competition – ethical considerations include fair competition and avoiding unethical practices like reverse engineering for commercial gain. 2. Intellectual property rights – respect the intellectual property rights of the original software creators. 3. Transparency – developers should be transparent about the methods used to create their software products.   **Potential risks:**   1. Intellectual property violation – reverse-engineering a software application can infringe on the original creators' intellectual property rights. 2. Legal consequences – the programmer may face legal actions, injunctions, or damages for IP violations. 3. Ethical scrutiny – engaging in questionable practices like reverse engineering can lead to ethical scrutiny and reputational damage.   **Strategies for ensuring compliance:**   1. Follow legal guidelines – ensure that reverse engineering is conducted within the bounds of legal guidelines and fair use principles and ensure compliance with IP laws. 2. Obtain permissions – seek permission or licences from the software creators before using reverse-engineered insights to develop competing products. 3. Focus on innovation – instead of copying existing software, focus on innovation and creating unique value propositions for the market. |

# Researching and planning

Table 7 – how apps can address challenges

|  |  |
| --- | --- |
| Challenge | Reason |
| Environmental challenges | Apps can promote environmental sustainability by providing tools for tracking carbon footprint, reducing waste and managing energy consumption.  Web-based software can facilitate online marketplaces for sustainable products, connect users with eco-friendly services and promote green initiatives. |
| Lifestyle challenges | Lifestyle apps can focus on health and wellness, offering fitness trackers, meal planning tools, mental health support and meditation guides.  Web software can provide educational resources on healthy living, personalised coaching services and community support networks for lifestyle improvement. |
| Societal challenges | Apps can address societal challenges by promoting inclusivity, diversity and social engagement through networking platforms, cultural awareness resources and community-building features.  Web-based software can facilitate online forums for discussions on societal issues, connect users with local resources and support groups and promote advocacy and activism. |
| Economic challenges | Apps can support economic empowerment by offering budgeting tools, financial management platforms, investment guidance and job search resources.  Web software can provide e-commerce platforms for small businesses, crowdfunding opportunities, financial education resources and access to microfinance services. |
| Cyber safety challenges | Apps and web software can enhance cyber safety by offering secure communication tools, encrypted messaging platforms, password managers and antivirus software.  Educational apps can provide cybersecurity training, tips for safe online behaviour and resources for protecting personal data and privacy online. |

## Understanding challenges in software development

Apps and web-based software can address various challenges in several ways listed in Table 8 below.

**Activity 14:** brainstorming app ideas

****In groups, students are to [brainstorm](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/542) different app ideas that fulfil the given scenario.

Table 8 – challenges, scenarios and ideas

|  |  |  |
| --- | --- | --- |
| Challenge | Scenario | Example ideas |
| Environmental challenges | Develop software solutions that promote environmental sustainability and encourage eco-friendly practices. | Recycling apps, energy-saving monitoring tools, sustainable transportation platforms and eco-friendly lifestyle trackers. |
| Societal inclusivity | Create software solutions that promote inclusivity, diversity and accessibility within society. | Diversity and inclusion training apps, accessibility tools for people with disabilities, cultural awareness platforms and community engagement apps. |
| Cyber safety and security | Design software solutions that enhance cyber safety, protect user data and prevent online threats. | Secure password managers, antivirus and anti-malware software, digital privacy protection tools and cybersecurity education platforms. |
| Economic challenges | Develop software solutions that support economic empowerment, financial literacy and entrepreneurship. | Financial management apps, budgeting tools, small business support platforms, microfinance and investment platforms. |
| Health and wellness challenges | Create software solutions that promote physical and mental health, fitness and wellbeing. | Health tracking apps, mental wellness platforms, telemedicine and virtual healthcare services, fitness and nutrition guides. |
| Community engagement | Develop software solutions that foster community engagement, social activism and civic participation. | Community organising platforms, social impact networks, voting and civic engagement apps, volunteering and donation platforms. |

## Object-oriented programming

Object-oriented (OOP) programming is a programming model that organises design around objects or data rather than functions and logic. OOP allows the developer to focus on the object they want to manipulate, rather than the logic required to manipulate them. This type of programming is suited to large, complex apps that are actively updated and maintained. This method is beneficial for collaborative development and for code reusability, scalability and efficiency.

**Activity 15:** object-oriented programming

1. Explain object-oriented programming.

|  |
| --- |
| **Sample answer:**  Object-oriented programming (OOP) is a way of organising and structuring code. It is based on the idea of objects, which are individual units of data and behaviour. Each object is an instance of a class, which is like a blueprint or template for creating objects |

1. List some examples of an object-oriented programming language in the space below.

|  |
| --- |
| **Sample answers include:**  C++  Java  C#  Python  Ruby  Swift. |

**Teacher note:** if you are unfamiliar with object-oriented programming principles in Python, utilise the [Object Oriented Programming: Getting Started for Australian Teachers](https://sites.google.com/dltv.vic.edu.au/object-oriented) resource. There are videos and activities that can be used to assist in explaining and for students to demonstrate their understanding.

**Activity 16:** features of object-oriented programming language

Complete Table 9 below with features of an object-oriented programming language.

Table 9 – features of an object-oriented programming language

|  |  |
| --- | --- |
| Feature | Use in object-oriented programming language |
| Classes | **Sample answer:**  The core concept of object-oriented programming is the class, which is a blueprint or template for creating objects. |
| Objects | **Sample answer:**  Objects are instances of a class and have their own properties and behaviour. |
| Abstraction | **Sample answer:**  Remove some characteristics from a class to have only necessary or essential properties and behaviours. |
| Encapsulation | **Sample answer:**  An object-oriented language allows for encapsulation, which means that the data and behaviour of an object can be hidden from the rest of the program and only exposed through a set of defined interfaces. |
| Inheritance | **Sample answer:**  Object-oriented languages support inheritance, which allows for the creation of new classes that inherit the properties and behaviour of existing classes. This allows for the reuse of existing code and the creation of a hierarchical class structure. |
| Polymorphism | **Sample answer:**  Object-oriented languages support polymorphism, which allows objects of different classes to be treated as objects of a common superclass. This allows for more flexible and reusable code. |

**Activity 17:** features of object-oriented programming language

****Using the game Space Invaders as reference, create a list of properties and behaviours (attributes and methods) for the main gameplay. Think about how the parts of the game function, for example the player has lives and can move left and right.

Table 10 – properties and behaviours of Space Invaders gameplay

|  |  |
| --- | --- |
| Space Invaders classes | Properties and behaviours |
| Player | Properties:   * image/design * lives * firing speed.   Behaviours:   * move * fire * die. |
| Aliens | Properties:   * image/design * lives.   Behaviours:   * move * shoot. |
| Barrier | Properties:   * image/design * lives/hit points.   Behaviours:   * barrier break down. |
| Bullet | Properties:   * design/colour * speed/power.   Behaviours:   * movement. |

## Elements and features of code

****As a class watch [Computer Science Basics: Sequences, Selections, and Loops (2:26)](https://www.youtube.com/watch?v=eSYeHlwDCNA).

While watching the video, complete the definitions and questions for standard control structures, including examples where possible.

Table 11 – definitions and questions for standard control structures

|  |  |
| --- | --- |
| Logic | Definition |
| Sequences | **Sample answer:**  Sequences are a set of instructions that are executed in a specific order, one after the other. |
| Selection (branching) | **Sample answer:**  Branching allows for the creation of different paths or outcomes based on conditions. |
| Iteration (loop) | **Sample answer:**  Iteration is the repeated execution of a certain set of instructions, which can be used to create loops or cycles in the game or simulation. |

**Activity 18:** sequence control structures and relational operators

1. What is the benefit of the sequence control structure when designing games?

|  |
| --- |
| **Sample answer:**  The benefit of the sequence control structure in game design is that it ensures actions and events occur in a specific, logical order, creating a predictable flow of gameplay. This allows game designers to structure tasks, events, or processes in a step-by-step manner, making it easier to control how the game progresses and ensuring that player interactions, animations and events happen in a coherent and consistent sequence. It also helps in debugging and modifying the game logic, as each step follows the previous one in a clear, linear progression. |

1. What are logical operators?

|  |
| --- |
| **Sample answer:**  Logical operators are used to combine multiple conditions and create more complex conditions. They are used to evaluate the truthfulness of certain statements and determine the outcome of certain conditions.  The 3 most common logical operators are:  AND (&& or and): the AND operator returns true if both of the conditions being compared are true.  OR (|| or or): the OR operator returns true if at least one of the conditions being compared is true.  NOT (! or not): the NOT operator negates the truthfulness of a statement. |

1. What are relational operators?

|  |
| --- |
| **Sample answer:**  Relational operators are used to compare values and determine the outcome of certain conditions. They are used to evaluate the relationship between 2 values and return a Boolean value (either true or false). The most common relational operators are:  Greater than (>)  The greater than operator returns true if the value on the left side of the operator is greater than the value on the right side.  Less than (<)  The less than operator returns true if the value on the left side of the operator is less than the value on the right side.  Greater than or equal to (>=)  The greater than or equal to operator returns true if the value on the left side of the operator is greater than or equal to the value on the right side.  Less than or equal to (<=)  The less than or equal to operator returns true if the value on the left side of the operator is less than or equal to the value on the right side.  Equal to (==)  The equal to operator returns true if the value on the left side of the operator is equal to the value on the right side.  Not equal to (!= or <>)  The not equal to operator returns true if the value on the left side of the operator is not equal to the value on the right side.  These relational operators are essential in making decisions in games and simulations and are commonly used in control structures such as if-else statements and loops, to determine the flow of the code. |

#### Practical application – learning through coding

Students could use online learning resources to learn and enforce the key elements and features of code. For example: [Grok Academy](https://groklearning.com/) or [w3schools](https://www.w3schools.com/python/).

The below explanations are generic but could be adapted to be in any other general-purpose programming language that is being used.

Table 12 – examples for understanding elements and features of code

|  |  |
| --- | --- |
| Elements and features of code | Explanation |
| Variables | * Storing and manipulating data, representing values or states within the program * Variable names (for example count, total\_amount), data types (integer, float, string), variable assignments (**=** operator) |
| Functions | * Encapsulating reusable code blocks, performing specific tasks or operations and modularising code * Function names (for example calculate\_area, print\_message), parameters (input arguments), function calls (depends on language chosen) |
| Conditional statements | * Controlling program flow based on conditions, making decisions and executing different code paths * Keywords (for example if, else, elif), comparison operators (e.g., ==, <, >), logical operators (and, or, not) |
| Loops | * Repeating code execution, iterating over data structures (lists, arrays), processing multiple items * Loop types (for example for, while), loop variables, loop conditions, loop body |
| Data structures | * Organising and storing data, accessing and manipulating data elements, representing collections of related values * Lists, arrays, dictionaries, sets, tuples |
| Classes and objects | * Object-oriented programming concepts, encapsulation, inheritance, polymorphism * Class definitions, object instances, attributes (properties), methods (behaviours) |
| Comments | * Providing explanatory notes, documenting code functionality and improving code readability * Comments (single-line # or multi-line ''' or """), documentation strings (docstrings) |
| Imports and libraries | * Features: Utilising external code and functionality, modularising codebase, leveraging existing libraries and frameworks * Import statements (import module\_name, from module\_name import function\_name), library functions and modules |
| File input/output | * Reading and writing data to files, file management, data persistence * File operations (open, read, write, close), file paths, file modes (r, w, a) |

**Activity 19:** identifying features in existing code

****Students identify the features of the code in each of the samples below. Depending on the general-purpose programming language chosen, the sample code could be changed.

Table 13 – features of code

|  |  |  |
| --- | --- | --- |
| Scenario | Features | Sample code example |
| Temperature converter | * Variables (for storing temperature values) * User input (to enter temperature values) * Conditional statements (to choose conversion direction) * Mathematical operations (to perform temperature conversion) * Output (displaying converted temperature) | Code sample for temperature converter: def celsius_to_fahrenheit(celsius):     return (celsius * 9/5) + 32  def fahrenheit_to_celsius(fahrenheit):     return (fahrenheit - 32) * 5/9  print("Temperature Converter") print("1. Celsius to Fahrenheit") print("2. Fahrenheit to Celsius")  choice = input("Enter your choice (1 or 2): ")  if choice == '1':     celsius = float(input("Enter temperature in Celsius: "))     converted_temp = celsius_to_fahrenheit(celsius)     print(f"{celsius}°C is equal to {converted_temp}°F") elif choice == '2':     fahrenheit = float(input("Enter temperature in Fahrenheit: "))     converted_temp = fahrenheit_to_celsius(fahrenheit)     print(f"{fahrenheit}°F is equal to {converted_temp}°C") else:     print("Invalid choice. Please enter 1 or 2.") |
| Guess the number game | * Random number generation * Use of a library * User input and validation (guessing a number) * Loop (to allow multiple guesses until the correct number is guessed) * Conditional statements (to check if the guess is correct) * Feedback messages (informing the player if their guess is too high or too low) | Code sample for Guess the number game:  import random  # Generate a random number between 1 and 100 secret_number = random.randint(1, 100) attempts = 0  print("Welcome to the Guess the Number Game!") print("I'm thinking of a number between 1 and 100.")  # Get user input and validate while True:     while True:         try:             guess = int(input("\nEnter your guess (1-100): "))             if 1 <= guess <= 100:                 break             else:                 print("Please enter a number between 1 and 100.")         except ValueError:             print("Please enter a valid number.")      attempts += 1      # Check if the guess is correct |

**Teacher note:** teachers may give a description and key features and ask students to write the code in Python.

Alternatively, teachers can provide the sample code and ask students to identify the key features, then move on to visually representing code through given features.

## The perspective of diverse groups

Students will be divided into teams to research how apps and web software are seen from the perspective of diverse groups. They will analyse the inclusivity of existing digital platforms and create a presentation that focuses on factors such as language support, cultural representation, accessibility and user interface design. Students will reflect on the importance of inclusivity in digital platforms and the impact of inclusive design on user experiences for their assigned group.

**Activity 20:** perspective of diverse groups research

****Considering the perspective of diverse groups when developing apps and web software is an important part of the design process.

By considering these perspectives, developers are creating inclusive products for all members of our community.

In this task, you will be working in small groups and assigned one of the following perspectives to consider:

* Aboriginal and Torres Strait Islander people
* culturally and linguistically diverse people
* people of different ages and gender
* people with disability.

1. In your groups, [brainstorm](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/542) potential perspectives they may need to consider for the group you have been assigned to research.
2. Research and identify factors that could contribute to the inclusivity of apps and websites for your specific group. For example, accessibility features for people with disability, avoiding cultural assumptions and considering the ownership of art works and images of or by Aboriginal and Torres Strait Islander people, considering the needs of older and younger users when developing an app.
3. Create a presentation that identifies the perspectives you have researched and reflects on the importance of inclusivity and its impact on the user experience.

|  |
| --- |
| **Guidance for the research task:**  Inclusive design principles  Key elements of inclusive design, such as accessibility, cultural sensitivity, language support and user interface considerations for different age groups and abilities.  Accessibility features  Explore the accessibility features present in apps and web software, focusing on functionalities that support users with disabilities, language translation options and features that cater to the specific cultural and age-related needs of diverse groups.  User experience analysis  Evaluating the ease of navigation, language options, visual representation and the presence of inclusive content and features.  Ethical considerations  Respecting cultural sensitivities, privacy and the diverse needs of potential users. |

Activity 21: brainstorming and ideation

As a class, generate potential ideas for new apps and web software concepts that prioritise inclusivity for diverse groups. Consider features, functionalities and design elements that would cater to the specific needs and perspectives of the diverse groups researched in Activity 20.

Each group will choose a concept that was brainstormed and creates a pitch for the new inclusive app or web software which communicates how this idea addresses the needs and perspectives of the diverse group, using thoughtful consideration of inclusivity.

Record your notes for the pitch in the space below.

|  |
| --- |
|  |

## Exploring the impact of app and web software on challenges

**Activity 22:** app case studies

You will be assigned to a group and given an app case study to investigate.

Read the assigned case study in Table 14 below. Then create a [concept map](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/577) that outlines how the app deals with environmental, lifestyle, societal and economic challenges and cyber safety considerations.

Table 14 – app case studies

|  |  |
| --- | --- |
| App case study | Possible points |
| EcoChallenge  EcoChallenge is a web and mobile app that promotes sustainability and environmental consciousness among individuals and communities. | 1. **Environmental challenge:** the app encourages users to take on eco-friendly challenges such as reducing waste, conserving energy and adopting sustainable lifestyle practices. 2. **Societal impact:** by fostering awareness and behaviour change, EcoChallenge contributes to a more sustainable society and promotes collective action for environmental preservation. 3. **Cyber safety measures:** the app prioritises user privacy and data security, employing encryption protocols and secure data storage to protect user information. 4. **Economic aspect:** while the app is free for individual users, it may generate revenue through partnerships with eco-friendly brands and organisations. |
| Headspace  Headspace is a meditation and mindfulness app designed to improve mental wellbeing and reduce stress. | 1. **Lifestyle challenge:** the app addresses the modern lifestyle challenge of stress management and mental health by offering guided meditation sessions and mindfulness exercises. 2. **Societal impact:** Headspace contributes to a healthier society by promoting mental wellbeing and resilience, thereby reducing the societal burden of stress-related illnesses. 3. **Cyber safety measures:** the app prioritises user privacy and confidentiality, implementing strict data protection measures and secure authentication methods. 4. **Economic aspect:** while the app offers a freemium model with basic features for free, it generates revenue through premium subscriptions and partnerships with healthcare providers. |
| Khan Academy  Khan Academy is an educational platform offering free online courses and resources across various subjects. | 1. **Economic challenge:** the platform addresses economic challenges related to education accessibility by providing free, high-quality educational content to learners worldwide. 2. **Societal impact:** Khan Academy empowers individuals from diverse backgrounds, contributing to a more educated and skilled society. 3. **Cyber safety measures:** the platform ensures data security and user privacy, adhering to strict guidelines for data encryption, secure login processes and protection against cyber threats. 4. **Environmental aspect:** while not directly related to environmental challenges, Khan Academy's online format reduces the environmental impact associated with traditional printed materials and commuting to physical classrooms. |
| Uber  Uber is a ride-hailing app that revolutionised the transportation industry by providing on-demand rides and promoting shared mobility. | 1. **Economic challenge:** Uber addresses economic challenges related to transportation affordability and accessibility by offering a cost-effective alternative to traditional taxis and private cars. 2. **Societal impact:** the app contributes to a more connected society by facilitating convenient and efficient transportation options, reducing reliance on personal vehicles and easing traffic congestion in urban areas. 3. **Cyber safety measures:** Uber prioritises user safety and security, implementing features such as GPS tracking, driver background checks, ride feedback mechanisms and emergency assistance options. 4. **Environmental aspect:** while not directly focused on environmental challenges, Uber's promotion of ride-sharing and efficient routing contributes to reduced emissions and environmental impact compared to individual car ownership and inefficient transportation methods. |

## Design principles and issues relevant to apps

**Activity 23:** design principles

Research the design principles listed in Table 15 below and describe how each principle applies to designing apps and web-based software.

Table 15 – design principles in app and web-based software

|  |  |
| --- | --- |
| Design principle | Description |
| User-centred design | **Sample answer:**  Designing apps with a focus on user needs, preferences and behaviours to create intuitive and engaging experiences. |
| Visual design | **Sample answer:**  Incorporating elements such as colour schemes, typography, icons and layout to enhance aesthetics and usability. |
| Navigation and information architecture | **Sample answer:**  Structuring the app's content and navigation paths logically to ensure easy and intuitive user navigation. |
| Accessibility | **Sample answer:**  Ensuring that the app is accessible to users with disabilities, including considerations for screen readers, colour contrast and alternative input methods. |
| Performance and optimisation | **Sample answer:**  Optimising app performance, load times and responsiveness across various devices and network conditions. |
| Security and privacy | **Sample answer:**  Implementing security measures to protect user data and privacy, including encryption, authentication and data handling practices. |

**Activity 24:** design challenge scenarios

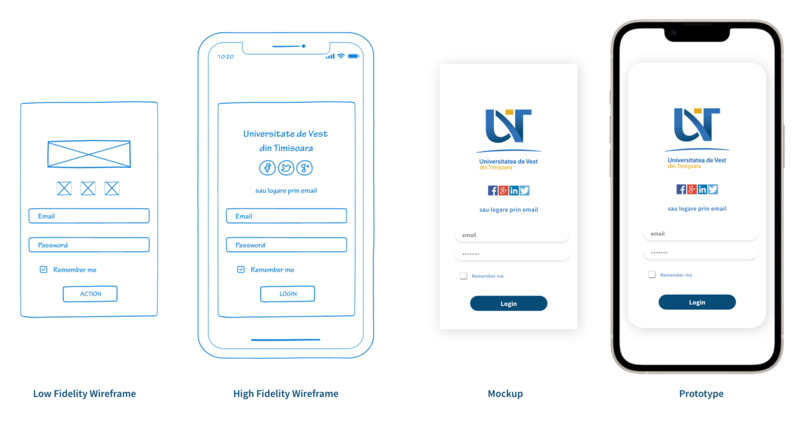
****As a class watch [How to create your first wireframe (8:34)](https://www.youtube.com/watch?v=KdfO_e0yK-g) and view examples of [wireframes](https://decode.agency/article/mobile-app-wireframe-examples/) to examine the different design options that can be incorporated into a design. To create a wireframe diagram, use software such as Microsoft Word, Canva, Adobe Express or Figma. As a class discuss, [What is wireframing?](https://www.figma.com/resource-library/what-is-wireframing/)

There are 8 different apps outlined in the design challenge, including:

|  |  |
| --- | --- |
| * health and fitness | * travel planning |
| * education | * finance management |
| * e-commerce | * entertainment streaming |
| * social networking | * environmental awareness. |

Choose one of the app ideas to sketch a wireframe or design for the app.

Figure 3 – wireframe diagrams



[‘Wireframe mockup prototype](https://commons.wikimedia.org/wiki/File:Wireframe_mockup_prototype.png)’ by [Alexandruionascu](https://commons.wikimedia.org/w/index.php?title=User:Alexandruionascu&action=edit&redlink=1),is licensed under [CC BY-SA 4.0](https://creativecommons.org/licenses/by-sa/4.0).

Students should consider in their wireframes the following:

* user story
* navigation
* basic visual design elements.

Table 16 – app 1

|  |  |  |
| --- | --- | --- |
| App 1 | Design challenge | Design principles to consider |
| Health and fitness app | Design an interface for a health and fitness app that tracks users' exercise routines, nutrition and progress towards fitness goals. | * **User-centred design:** personalised workout plans based on user goals and fitness levels, intuitive navigation for easy access to features like tracking workouts and nutrition * **Visual design:** clear and visually appealing UI elements for displaying exercise routines, progress charts and nutritional information, colour coding for different types of exercises or food categories * **Navigation and information architecture:** categorised sections for workouts, nutrition tracking, progress logs and settings, with a hierarchical structure for easy navigation between pages * **Accessibility:** adjustable font sizes, high contrast UI for readability, voice-guided workout instructions for users with visual impairments * **Performance and optimisation:** fast loading times for workout videos and nutrition data, optimising battery usage during workout tracking * **Security and privacy:** secure login with two-factor authentication and encrypted data storage for user health information |

Table 17 – app 2

|  |  |  |
| --- | --- | --- |
| App 2 | Design challenge | Design principles to consider |
| Education app for students | Create a user-friendly interface for an education app targeted at students, including features for accessing course materials, submitting assignments and communicating with instructors. | * **User-centred design:** student dashboard with course schedules, assignment deadlines and grades, intuitive navigation for accessing course materials and discussions * **Visual design:** organised layout for course content, interactive elements like quizzes and multimedia resources, consistent colour scheme and branding * **Navigation and information architecture:** course hierarchy with modules, lessons and assessments, search functionality for finding specific topics or materials * **Accessibility:** screen reader compatibility, text-to-speech options for course content, adjustable contrast for readability * **Performance and optimisation:** quick loading of course materials and multimedia content, offline access for studying on the go * **Security and privacy:** secure login for students and instructors, permissions management for accessing course content and grades |

Table 18 – app 3

|  |  |  |
| --- | --- | --- |
| App 3 | Design challenge | Design principles to consider |
| e-commerce app for local businesses | Design a shopping app specifically for local businesses, allowing users to browse products, make purchases and support local vendors. | * **User-centred design:** personalised user experience based on purchase history, preferences and browsing patterns. Intuitive navigation for quickly discovering and purchasing products, with features like quick search, wishlists and personalised recommendations. * **Visual design:** clear and visually appealing UI elements to display product listings, vendor information and promotions. * **Navigation and information architecture:** categorised sections for product browsing, vendor information, order tracking, and user settings. Ensure a hierarchical structure for easy navigation between product categories and purchase processes. * **Accessibility:** adjustable font sizes, high-contrast UI for readability, and voice search for users with visual impairments. Include options like screen reader compatibility and alternative text for product images to ensure usability for all. * **Performance and optimisation:** fast loading times for product images, vendor pages and payment processing. Optimise the app for efficient battery usage, especially during browsing and order placement. * **Security and privacy:** secure login options with two-factor authentication, encrypted storage for user payment and personal information, and secure transaction handling. |

Table 19 – app 4

|  |  |  |
| --- | --- | --- |
| App 4 | Design challenge | Design principles to consider |
| Social networking app for community engagement | Develop a social networking app that promotes community engagement, events and discussions among users with shared interests or geographic locations. | * **User-centred design:** customisable user profiles, news feed algorithms based on user interests and interactions, easy sharing and commenting features * **Visual design:** engaging visual content such as images and videos in the news feed, consistent design elements for profile pages and posts * **Navigation and information architecture:** clear navigation tabs for home, profile, notifications and messaging, content filtering options for personalised feed * **Accessibility:** captioning for multimedia content, text resizing options, voice commands for navigation * **Performance and optimisation:** efficient content caching for faster load times and push notifications for real-time updates * **Security and privacy:** privacy settings for controlling profile visibility and post audience, content moderation features and secure messaging with end-to-end encryption |

Table 20 – app 5

|  |  |  |
| --- | --- | --- |
| App 5 | Design challenge | Design principles to consider |
| Travel planning app | Create an app interface for planning and organising travel itineraries, including features for booking flights, accommodations and activities, as well as providing travel tips and recommendations. | * **User-centred design:** customisable travel itineraries based on user preferences, interactive maps with points of interest and directions, real-time flight and accommodation booking * **Visual design:** inspirational travel imagery, intuitive booking forms with clear pricing and availability information and map overlays for trip routes * **Navigation and information architecture:** structured itinerary sections for flights, accommodations, activities and dining, filtering options for refining search results * **Accessibility:** voice-guided navigation for directions, alt text for map images, screen reader support for itinerary details * **Performance and optimisation:** quick search and booking functionalities, offline access to travel plans, low data usage for map and itinerary updates * **Security and privacy:** secure payment processing. |

Table 21 – app 6

|  |  |  |
| --- | --- | --- |
| App 6 | Design challenge | Design principles to consider |
| Finance management app | Design a finance management app that helps users track expenses, manage budgets and set financial goals, with features for expense categorisation, bill reminders and financial insights. | * **User-centred design:** customisable budgeting tools based on user income and expenses, interactive expense tracking with categorisation and analysis * **Visual design:** clear graphs and charts for financial insights, colour-coded expense categories, intuitive dashboard for budget overview * **Navigation and information architecture:** sections for income, expenses, savings goals and reports, drill-down options for detailed transaction views * **Accessibility:** voice commands for expense input, high contrast mode for visibility, screen reader compatibility for financial data * **Performance and optimisation:** fast data syncing across devices, offline mode for expense logging, automated alerts for budget milestones * **Security and privacy:** strong encryption for financial data storage, secure authentication methods, privacy controls for data sharing and analytics |

Table 22 – app 7

|  |  |  |
| --- | --- | --- |
| App 7 | Design challenge | Design principles to consider |
| Entertainment streaming app | Develop an interface for an entertainment streaming app that offers a wide range of content such as movies, TV shows, music and podcasts, with personalised recommendations and user profiles. | * **User-centred design:** personalised content recommendations based on user viewing history and preferences, easy content discovery and browsing * **Visual design:** engaging media player interface with playback controls, curated playlists and collections and visually appealing artwork for movies and music * **Navigation and information architecture:** sections for genres, trending content, user playlists, search filters and detailed metadata for media content * **Accessibility:** audio descriptions for visually impaired users, subtitle customisation options and keyboard shortcuts for media controls * **Performance and optimisation:** adaptive streaming quality based on network conditions and offline downloads for content viewing without the internet * **Security and privacy:** secure user profiles with password protection, parental controls for content restrictions and encryption for payment transactions |

Table 23 – app 8

|  |  |  |
| --- | --- | --- |
| App 8 | Design challenge | Design principles to consider |
| Environmental awareness app | Create an app interface focused on raising environmental awareness, featuring educational content, tips for eco-friendly living, local environmental events and ways for users to take action for sustainability. | * **User-centred design:** educational content tailored to user interests and learning preferences, interactive quizzes and challenges for eco-friendly practices * **Visual design:** infographics and interactive visuals for environmental data, gamification elements for engaging user participation * **Navigation and information architecture:** sections for eco-tips, local environmental initiatives, climate data and user contributions and search functionality for specific topics * **Accessibility:** text-to-speech options for content consumption, alt text for images and graphs and colourblind-friendly design elements * **Performance and optimisation:** fast content loading for quick access to environmental news and updates, data compression for minimal data usage * **Security and privacy:** secure user accounts with privacy settings, data encryption for user contributions and transparent data handling policies |

**Teacher note:** this activity could be completed in pairs, with the teacher assigning a certain scenario to groups. Additionally, other scenarios could be chosen depending on the demographic of the class. After completing a self-evaluation, students could also attempt multiple design challenges.

## Self-assessment

Tick the appropriate box for each of the learning intentions.

1 = I need a challenge, I can teach this to others.

2 = I can do this confidently.

3 = I can do this, but need more practice and revision.

4 = I can do this with some assistance.

Table 24 – self-assessment of learning intentions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Learning intentions | 1 | 2 | 3 | 4 |
| Break down a user interface into its basic steps so that they can be applied to a digital solution. |  |  |  |  |
| Determine the user interface needed to complete each step of the interactive media product. |  |  |  |  |
| Plan how each user interface component will look. |  |  |  |  |
| Design a wireframe. |  |  |  |  |
| Represent the user interface in an interactive media product. |  |  |  |  |
| Evaluate a peer’s product based on their choice of user interface. |  |  |  |  |
| Test and evaluate a UI to improve overall efficiency. |  |  |  |  |
| Evaluate a peer’s product. |  |  |  |  |

## Evaluate whether solutions meet specific requirements

**Activity 25:** evaluate a solution against a criteria

Students will use their prior learning of social and ethical considerations, legal responsibilities and cybersecurity principles to evaluate a created solution.

Students will look at an existing app and evaluate it against the criteria outlined in Table 25 below. Some of the criteria may not apply to the solutions students are evaluating. As they conduct their evaluation, students can make suggestions on how they think the app could be adapted to meet the criteria.

Table 25 – social and ethical considerations, legal responsibilities and cybersecurity principles

|  |  |
| --- | --- |
| Responsibilities | Criteria |
| Social impact | * Does the app promote inclusivity and diversity, respecting users' cultural backgrounds, beliefs and values? * Does the app contribute positively to societal wellbeing, addressing social issues or promoting social responsibility? * Are there features in the app that ensure user safety, wellbeing and mental health? * Does the app encourage ethical behaviour, responsible content sharing and respectful communication among users? |
| Ethical considerations | * Does the app have clear and transparent privacy policies regarding data collection, storage and usage? * Does the app obtain informed consent from users before collecting their personal information? * Are there mechanisms in place to protect users' confidentiality and anonymity, especially for sensitive data? * Does the app avoid deceptive or manipulative practices, such as misleading advertisements or hidden fees? * Does the app comply with ethical guidelines and industry standards for software development? |
| Legal responsibilities | * Is the app compliant with relevant laws and regulations, such as data protection laws? * Does the app respect intellectual property rights, including copyrights, trademarks and patents? * Are there terms of service and user agreements that outline users' rights and responsibilities? * Does the app have mechanisms for reporting and addressing legal issues, such as copyright infringement or cyberbullying? |
| Cybersecurity principles | * Does the app use secure authentication mechanisms, such as strong passwords, multi-factor authentication or biometrics? * Is data encryption implemented to protect sensitive information during transmission and storage? * Are there measures to prevent unauthorised access, data breaches and cyberattacks? * Does the app regularly update security patches and software versions to mitigate vulnerabilities? * Are there protocols in place for incident response, data breach notification and user data recovery in case of security incidents? |
| Overall user experience | * Is the app user-friendly, intuitive and accessible to users from diverse backgrounds and abilities? * Does the app provide adequate support and guidance for users to navigate its features and functionalities? * Are there options for users to provide feedback, report issues and request assistance within the app? * Does the app prioritise user satisfaction, engagement and trust through responsive design and timely updates? |

**Teacher note:** this activity could be completed after the Assessment task and students can use a peers' work to evaluate their own.

## Data privacy and cybersecurity in software development

**Activity 26:** Why is data privacy and cybersecurity important in software development?

****Using the [5 whys](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/638) template, provide 5 reasons why it is important to have data privacy and cybersecurity in today’s digital world, considering personal concerns about sharing information online.

|  |
| --- |
| **Sample discussion points:**   * Protection of personal information. Safeguarding personal data such as names, addresses and financial details is essential to prevent identity theft and fraud. * Preservation of individual rights. Data privacy ensures that individuals have control over their personal information, respecting their right to privacy. * Prevention of unauthorised access. This helps in preventing unauthorised access to sensitive data, reducing the risk of data breaches. * Trust and reputation. Businesses and organisations that prioritise data privacy earn the trust of customers and stakeholders, enhancing their reputation. * Protection against cyber threats. Cybersecurity measures defend against a wide range of threats, including malware, ransomware and phishing attacks. * Preservation of data integrity. This ensures the accuracy and reliability of data, preventing unauthorised changes or deletions. * Protection of intellectual property. This safeguards intellectual property, trade secrets and proprietary information from theft and espionage. * Global connectivity. In an interconnected world, cybersecurity is vital for maintaining secure communications and collaborations across borders. * Personal safety. In the era of smart devices, cybersecurity is essential for preventing unauthorised access to connected vehicles, medical devices and home automation systems. * Ethical and legal obligations. Cybersecurity compliance is required by various laws and regulations to protect individuals' and organisations' digital assets. |

It is important to note that not all apps and programs collect the same data and users should carefully review an app's privacy policy or terms of service before downloading and using it to understand what data is being collected and how it will be used.

**Activity 27:** data collection

How do apps and web-based software developers collect data?

|  |
| --- |
| **Sample answers:**  **User input and forms:** apps and websites often collect data directly from users through input fields, forms and surveys. This data can include personal information (name, email, address), preferences, feedback and other user-generated content.  **Cookies and tracking technologies:** web software can use cookies, tracking pixels and similar technologies to gather information about user interactions, browsing behaviour and preferences. This data helps in personalising content, improving user experience and analysing website performance.  **Device information:** apps can access device-specific information such as device type, operating system, hardware specifications and location data (GPS coordinates). This data is used to optimise app functionality, deliver location-based services and analyse user demographics.  **Analytics tools:** software developers integrate analytics tools like Google Analytics or custom analytics solutions to track user interactions, app usage patterns and performance metrics. These tools provide insights into user behaviour, app performance and user engagement.  **Social media integration:** apps and websites often integrate with social media platforms (for example Facebook, X and Instagram) to collect data from user profiles, social interactions and shared content. This data can be used for targeted advertising, social sharing features and user authentication.  **Surveys and feedback mechanisms:** apps and websites may include surveys, feedback forms, or rating systems to gather user feedback, opinions and satisfaction levels. This qualitative data helps in improving product features and user experience.  **Transactional data:** e-commerce apps and websites collect transactional data related to purchases, orders, payments and shipping information. This data is essential for processing transactions, managing inventory and providing customer support.  **Usage logs and error reports:** apps and software collect usage logs, error reports and crash data to identify bugs, performance issues and user experience problems. Developers use this data for troubleshooting, debugging and optimising software performance.  **Consent and permissions:** it isimportant to note that data collection should always be done with user consent and in compliance with privacy regulations. Apps and websites typically request user consent for data collection and provide options for users to manage their privacy settings. |

**Activity 28:** Think, Pair, Share

Read Table 26 below, then in pairs, complete a [Think, Pair, Share](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/645) activity on the following question.

Why do app developers collect data from users of their apps and/or web software?

Table 26 – reasons for collecting data

|  |  |
| --- | --- |
| Reason | Explanation |
| User personalisation | By collecting data about user preferences, behaviour and interactions, developers can personalise the app experience for each user.  This includes recommending relevant content, suggesting personalised products or services and customising app features to match user preferences. |
| Improving user experience | Data collection helps developers understand how users navigate the app, which features they use most frequently and where they encounter challenges.  This information is valuable for optimising app design, streamlining user workflows and addressing usability issues to enhance overall user experience. |
| Performance optimisation | Data collection enables developers to monitor app performance metrics such as app load times, crash rates, error logs and user engagement metrics.  Analysing this data helps identify performance bottlenecks, optimise app speed and stability and ensure smooth functioning across different devices and platforms. |
| Analytics and insights | Gathering data allows developers to gain valuable insights into user demographics, usage patterns, market trends and app performance metrics.  These analytics help in making data-driven decisions, identifying growth opportunities and refining app strategies for better user acquisition and retention. |
| Product development and iteration | Data collected from user feedback, surveys and usage patterns informs product development cycles.  Developers can prioritise feature enhancements, bug fixes and new functionalities based on user feedback and data insights, leading to iterative improvements and continuous innovation. |
| Targeted advertising and marketing | Data about user preferences, behaviour, and demographics can be used for targeted advertising and marketing campaigns.  Developers can create personalised marketing messages, promotions and improve marketing effectiveness. |
| Monetisation strategies | For apps that rely on collecting money through ads, in-app purchases, subscriptions or other revenue streams, data collection plays a crucial role.  Understanding user behaviour and preferences helps in optimising monetisation strategies, pricing models and ad placements to maximise revenue generation. |
| Compliance and security | Collecting data also enables developers to ensure compliance with legal and regulatory requirements related to data privacy, security and user consent.  Developers must adhere to data protection laws and implement security measures to safeguard user data from unauthorised access or breaches |

**Activity 29:** data ownership in app development case study

****In the development of a mobile application called ’FitTrack’, which tracks users' fitness activities and provides personalised health recommendations, data ownership is a critical aspect that requires careful consideration.

**Data collection**

* FitTrack collects various types of data, including user profiles (age, gender, weight), fitness activity logs (steps taken, calories burned), GPS data (location for mapping activities) and health goals (weight loss, muscle gain).
* Data collection methods include user input during account creation, sensor data from smartphones (for example step counters) and optional integrations with wearable fitness devices.

**Ownership of data**

* The ownership of data in FitTrack is clearly outlined in the app's privacy policy. Users retain ownership of their personal data and fitness information. FitTrack and its parent company, FitTech, have limited rights to use this data strictly for app functionality and service improvements.
* FitTrack does not sell or share user data with third parties without explicit consent, except for anonymised and aggregated data used for analytics and research purposes.

**Data protection measures**

* FitTrack employs robust data protection measures to safeguard user data. This includes encryption of data in transit and at rest, secure authentication mechanisms and regular security audits.
* User data is stored securely on servers hosted by a reputable cloud service provider that complies with industry security standards and regulations.

**Privacy policies and user consent**

* FitTrack's privacy policy is transparent and easily accessible within the app. It provides detailed information on the types of data collected, how it is used and user rights regarding data access, modification and deletion.
* During the onboarding process, users are required to review and accept the privacy policy and consent to data collection and processing practices.

**Compliance and legal Requirements**

* FitTrack complies with relevant privacy laws and regulations.
* The app includes features for users to manage their privacy settings, opt-out of data sharing and request data exports or deletions as per legal requirements.

**Data interpretation and use**

* FitTrack interprets user data to provide personalised health insights, activity recommendations and progress tracking. However, data interpretation is done in a privacy-conscious manner, avoiding profiling or discriminatory practices.
* User consent is sought for specific data uses, such as sharing fitness achievements on social media or participating in health challenges with other users.

****Using the case study on FitTrack, answer questions from Table 27 below.

Table 27 – FitTrack case study

|  |  |
| --- | --- |
| Question | Possible answers |
| What types of data does FitTrack collect from users?  Can you provide specific examples of the data categories? | FitTrack collects:   * user profiles (age, gender, weight) * fitness activity logs (steps taken, calories burned) * GPS data (location for mapping activities) * health goals (weight loss, muscle gain). |
| How does FitTrack ensure that user data is collected and stored securely?  What encryption methods or security measures are in place? | FitTrack employs encryption of data in transit and at rest, secure authentication mechanisms and regular security audits to safeguard user data. |
| Who owns the data collected by FitTrack?  Are there any restrictions or limitations on how the data can be used? | Users own their personal data and fitness information, with FitTrack having limited rights to use it strictly for app functionality and service improvements. Data usage is governed by user consent and privacy policies. |
| Can users access, modify, or delete their data stored by FitTrack?  How is user consent obtained for data collection and processing? | FitTrack allows users to manage their data through privacy settings, opt-out of data sharing and request data exports or deletions as per legal requirements and user preferences. |
| Are there any third parties or service providers involved in data processing or storage for FitTrack?  If so, how does FitTrack ensure data confidentiality and security in these partnerships? | FitTrack partners with reputable cloud service providers that comply with security standards and regulations, ensuring data confidentiality and security in data processing and storage partnerships. |
| How does FitTrack interpret and analyse user data to improve user experience or provide personalised services?  Are there any ethical considerations in data interpretation? | FitTrack interprets user data for personalised health insights and recommendations, prioritising ethical data interpretation and avoiding discriminatory practices. |
| What measures does FitTrack have in place to detect and respond to potential cybersecurity threats, such as data breaches or malicious attacks? | FitTrack has cybersecurity measures such as intrusion detection systems, regular security audits and incident response protocols to detect and respond to cybersecurity threats. |
| Does FitTrack provide transparency about its data practices, including data sharing with third parties or data retention policies? | FitTrack's privacy policy and terms of use provide transparency on data practices, including data sharing, retention and user rights, ensuring transparency and informed user consent. |
| How does FitTrack address user concerns or inquiries related to data privacy and security?  Is there a dedicated privacy policy or support channel for users to seek assistance? | FitTrack has a dedicated privacy policy accessible within the app and users can contact support for inquiries or assistance regarding data privacy and security concern. |

****Assess FitTrack's adherence to privacy and cybersecurity principles in data collection, ownership, protection and interpretation. Use the case study and the related questions to help form your answer using the [PEEL paragraph writing technique](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/625).

Assess means to make a **judgement of value**, quality, outcomes, results or size. In this answer, you will be making a judgement of the value of FitTrack’s adherence to privacy and cybersecurity principles.

## Data protection and cybersecurity

**Activity 30:** cybersecurity in 7 minutes

****As a class, watch [[What is Cyber Security? | How it works? (7:06)](https://www.youtube.com/watch?v=inWWhr5tnEA).](https://www.youtube.com/watch?v=inWWhr5tnEA)

Companies and individuals must be aware of a variety of cyber-attacks. After watching the video,identify the different cyber-attacks that could target companies and individuals.

|  |
| --- |
| **Sample answers:**   * Malware attack * Phishing attack * Man-in-the-middle attack * Password attack * Cybersecurity practices * Impact of a cyber attack * Advanced Persistent Threat (APT) * Denial of service attack and distributed denial of service (DDoS) attack * SQL injection attack |

**Key areas of data protection and cybersecurity**

**Encryption**

The process of converting data into a code to prevent unauthorised access. It acts as a secure envelope around sensitive data, ensuring that even if it's intercepted, it remains confidential. In gaming, encryption is crucial for safeguarding sensitive player data, such as login credentials, payment information and personal details. This ensures that even if a cybercriminal gains access to the data, they cannot decipher it without the encryption key.

**Authentication**

The verification process to ensure that users are who they claim to be. This includes strong passwords and multi-factor authentication which reduces the impact of a data breach and increases a user’s defence against phishing.

**Secure storage**

Storing data in a way that safeguards it from unauthorised access or data breaches. Some secure data storage practices and technologies include data encryption, access control, regular backups, data retention policies and employee training.

Safeguarding user data against breaches and unauthorised access is important. This is because:

* protecting user data builds trust and player confidence in the game and the game company
* data breaches can result in financial losses, damage to reputation and legal consequences for game companies
* unauthorised access to personal information can lead to identity theft and other privacy violations for players.

**Activity 31:** real-world data breaches in the software development industry

****Research the following real-world data breaches in the software development industry. Provide an overview of what happened and discuss the consequences for both the user and the developer or company.

Table 28 – real-world data breaches in the software development industry

|  |  |  |
| --- | --- | --- |
| Breach | Overview | Consequence |
| Facebook-Cambridge Analytica data scandal (2018) | In 2018, it was revealed that the political consulting firm Cambridge Analytica harvested data from millions of Facebook users without their consent. This data was used for targeted political advertising during elections. | **Consequences for users:**   * users' personal data, including profile information, likes and interactions were accessed without their knowledge or permission * this breach compromised user privacy and raised concerns about data misuse.   **Consequences for the developer or company:**   * Facebook faced severe backlash and regulatory scrutiny for its data handling practices * Facebook’s reputation was tarnished, leading to trust issues among users and investors * Facebook implemented stricter data policies and faced fines and legal actions from regulators. |
| Equifax data breach (2017) | In 2017, Equifax, a major credit reporting agency, experienced a data breach where hackers gained access to sensitive personal information of over 147 million consumers, including social security numbers, birthdates and credit card details. | **Consequences for users:**   * the Equifax breach exposed users to identity theft, financial fraud and unauthorised access to their credit information * many individuals faced financial losses and struggled to regain control of their compromised data.   **Consequences for the developer or company:**   * Equifax faced intense criticism for its lax security practices and delayed response to the breach * Equifax's stock value plummeted, and it incurred significant financial losses due to legal settlements, fines and remediation efforts. |
| SolarWinds data breach (2020) | In late 2020, SolarWinds, a software company providing network management tools, experienced a significant data breach. Hackers infiltrated SolarWinds' software update system and inserted malicious code into updates, allowing them to access the networks of numerous SolarWinds customers, including government agencies and major corporations. | **Consequences for users:**   * the SolarWinds breach compromised sensitive data and network infrastructure of government agencies, businesses and organisations worldwide * users' confidential information, communications and operationalsystems were potentially exposed, leading to security risks and vulnerabilities.   **Consequences for the developer or company:**   * SolarWinds faced severe reputational damage and legal repercussions for the breach * SolarWinds' stock value dropped, and it underwent extensive investigations, audits and remediation efforts * SolarWinds also faced lawsuits and regulatory penalties for failing to prevent the breach and protect customer data. |
| Ubiquiti Networks data breach (2021) | In early 2021, Ubiquiti Networks, a manufacturer of networking devices and software, disclosed a data breach where unauthorised access was gained to its IT systems.  The breach exposed customer information, including account credentials, but Ubiquiti initially downplayed the severity of the incident. | **Consequences for users:**   * Ubiquiti customers faced potential risks of account compromise, unauthorised access to network devices and data theft * the breach undermined user trust in Ubiquiti's security measures and raised concerns about data privacy and protection.   **Consequences for the developer or company:**   * Ubiquiti's handling of the breach, including its delayed disclosure and downplaying of the impact, led to criticism and scrutiny * Ubiquiti faced backlash from customers, regulatory scrutiny and legal challenges related to data protection compliance * Ubiquiti also implemented enhanced security measures and communication protocols in response to the incident. |

## Representing data

In programming, variables are used to store and manipulate data, such as numbers, text, or other types of information. It represents a symbolic name or label used to store and manage data within a program or equation.

Variables have names (identifiers) that programmers assign to them, making it easier to reference and modify the data they contain. Variables can store a range of data types including Boolean, characters, strings, integers and floats.

**Activity 32:** What are data types?

****As a class watch [What are data types? (3:52)](https://www.youtube.com/watch?v=A37-3lflh8I).

As students watch the videos they complete the [+1 routine](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/661). After the students have watched the video, they recall what the video was about. They then pass their worksheet to the next person, who elaborates on a topic, adds something new or makes a connection. Repeat this at least twice. Students then review their original responses and any new additions or elaborations made by their peers. Finally, students reflect, asking questions such as:

* What did you find as your read the ideas of others?
* How did it help you building on the thinking of others?
* How did it help you build your understanding of this topic?

**Teacher note:** this activity can be conducted as a paper activity or a virtual jamboard or whiteboard. Templates are available through the [Digital Learning Selector.](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Browser?clearCache=35de6a1-aa57-753c-b59f-eb5f20a84490)

## Data types

In programming, data types are essential because they define the kind of data that can be stored and manipulated within a program. The primary data types include:

* integers
* floats
* characters
* strings
* Boolean operators.

Each programming language may have unique sets of data types, but the primary data types are fundamental across most programming languages to ensure data is used efficiently and correctly.

**Activity 33:** benefits and limitations of data types

****As a class watch [Why TRUE + TRUE = 2: Data Types](https://www.youtube.com/watch?v=6otW6OXjR8c) (8:08) and discuss the benefits and limitations of data types.

****Use this information to complete Table 29 below by defining and outlining benefits and limitations of each listed data type. You can also research these terms.

Table 29 – benefits and limitations of data types

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | Definition | Benefits | Limitations |
| Boolean |  |  |  |
| Character |  |  |  |
| String |  |  |  |
| Integer |  |  |  |
| Float |  |  |  |

**Sample answer:**

|  |  |  |  |
| --- | --- | --- | --- |
| Data type | Definition | Benefits | Limitations |
| Boolean | A Boolean result can only have one of 2 possible values – true or false. Boolean operators are ‘AND’, ‘OR’ AND ‘NOT’. | Simple – easy way to manage conditions and control the flow of a program.  Clear – using Booleans can make the logic of a program easy to understand and read.  Efficient – Booleans are generally fast and require minimal memory.  Reduces errors – by using Boolean values, the risk of errors in conditional statements can be reduced. | Limited expressions – Booleans can only represent 2 states (true or false) and may not be sufficient for more complex conditions.  Binary – the simple nature of Booleans can be too simple for some programs where more detail is required.  Dependent on context – the interpretation of true or false can depend on the context leading to potential confusion if not clearly documented. |
| Character | Characters in programming represent individual letters, digits, symbols or punctuation marks and are the building blocks for strings. Each character is stored as a small piece of data usually following a standardised encoding system like ASCII or Unicode. | Text manipulation – characters are the foundations of strings, allowing programmers to manipulate text efficiently.  Simple – handling individual characters can simplify programming tasks.  Versatile – characters can represent more than just letters and numbers, enabling diverse applications. | Limited data – a single character only represents one symbol, which can be restrictive when dealing with more complex data.  Encoding issues – different systems and languages may use various character encodings, leading to potential compatibility issues.  Memory usage – while individual characters are small, manipulating large amounts of text at the character level can be inefficient in terms of memory. |
| String | A sequence of characters (letters, numbers or symbols) used to represent text in programming. | Text handling – strings allow programmers to store, manipulate and display text, allowing them to create user interfaces, process data and communicate with users.  Versatile – strings can be used in various applications and are a fundamental part of many programming tasks.  Built in functions when working with strings such as finding the length of a string, concatenating (joining) strings and searching for sub-strings.  Readability – using strings can make code easier to read and understand. | Memory usage – strings consume a significant amount or memory, especially when dealing large amounts of text.  Immutability – this means that strings cannot be changed after they are created. Any changes create a new string.  Performance – manipulating strings, especially large ones can be slower compared to other data types.  Complexity – working with strings can become complex when dealing with special characters, different encodings and localisations. |
| Integer | Integers are whole numbers without fractions or decimal parts and can be positive, negative and zero. | Simple – integers are straightforward and easy to understand.  Efficient – operations with integers such as addition, subtraction or multiplication  Range of uses – integers are widely used in a variety of applications such as counting items, managing loops and indexing arrays.  Precision – since integers don’t have fractional parts, they avoid rounding errors that can occur with floats. | No fractions or decimals – integers can’t represent fraction or decimal values which limits their use in scenarios using precise measurements or calculations.  Overflow – integers can have a maximum or minimum value that they can represent. If calculations go over these limits, it can lead to overflow and unexpected errors or results.  Fixed size – the range of integers is limited by the number or bits to store them. For example, a 32-bit integer has a specific range, and using larger integers requires more memory and is slower to process. |
| Float | A float or floating-point numbers are used in programming to represent numbers that have a decimal point. They can include very large and small numbers, allowing for a wide range of values. | Precision – floats are essential for calculations that require decimal points, such as scientific or financial calculations.  Wide range – floats can represent a much larger range of values than integers, including very small fractions and very large numbers.  Flexible – floats are useful in situations where the exact value is not a whole number  Built in functions – most programming languages offer a range of built in functions to manipulate and work with floats. These functions make tasks such as rounding or truncating simpler. | Precision issues – floats are not always perfectly accurate due to the way they are stored. This can lead to rounding errors in calculations.  Performance – calculations involving floats can be slower than those with integers due to their complexity.  Memory usage – floats generally take up more memory than integers because they store more information.  Complexity – working with floats can be more complex than working with integers especially when rounding or working precisely. This can make program debugging challenging. |

# Producing and implementing

## Flowcharts

Software developers use flowcharts to show the sequence of logical steps of a program. Flowcharts use simple shapes to show the process, and arrows show the relationship and process or data flow. View [Flowchart Elements](https://www.tutorialspoint.com/programming_methodologies/programming_methodologies_flowchart_elements.htm) to examine the various elements of a flow chart and their uses.

**Activity 34:** flowchart symbols and their purpose

Complete Table 30 below using the [Flowchart Elements](https://www.tutorialspoint.com/programming_methodologies/programming_methodologies_flowchart_elements.htm) weblink.

Table 30 – flowchart symbols and their purpose

|  |  |  |
| --- | --- | --- |
| Symbol | Symbol name | Purpose |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Teacher note:** alternative resources for this activity include [Guide to Flowchart Symbols, from Basic to Advanced](https://www.gliffy.com/blog/guide-to-flowchart-symbols), [Flowchart Symbols and Notation](https://www.lucidchart.com/pages/flowchart-symbols-meaning-explained) and [Flowchart Symbols - A Complete Guide](https://www.zenflowchart.com/flowchart-symbols).

**Sample answers:**

|  |  |  |
| --- | --- | --- |
| Symbol | Symbol name | Purpose |
|  | Start/stop | Use at the beginning and the end of an algorithm to show the start and finish of a program. |
|  | Process | Shows processes such as mathematical operations. |
|  | Input/output | Shows program inputs and outputs. |
|  | Decision | Highlights decision statements in a program. For example, yes or no. |
|  | Arrow | Shows the relationship between the shapes. |

## Pseudocode

Pseudocode is a way to express algorithms or program logic using structured natural language that resembles a programming language but is not bound by syntax rules. Here are some key features of pseudocode:

* pseudocode follows a structured format with indentation and logical flow similar to programming languages
* pseudocode is designed to be easily readable and understandable by both programmers and non-programmers
* pseudocode does not adhere to the syntax rules of any specific programming language, allowing for flexibility and clarity
* you can use variables and data types (such as integers, strings and arrays) in pseudocode to represent data and operations
* pseudocode includes conditional statements like if-else, switch-case and loops (for, while) to control the flow of execution
* you can define functions or procedures in pseudocode to encapsulate reusable code blocks
* pseudocode allows for comments to explain the logic or provide additional context within the code
* pseudocode may include error-handling logic using try-catch or error-checking conditions.

**Activity 35:** writing pseudocode

Complete Table 31 on the primary constructs of pseudocode using information provided by the teacher or from various websites.

Table 31 – primary constructs of pseudocode

|  |  |
| --- | --- |
| Primary constructs | How to write |
| Sequence | process 1  process 2  …  …  process n |
| Binary selection | 1. IF condition THEN  process 1  ENDIF  2. IF condition THEN  process 2  ELSE  process 1  ENDIF |
| Multiway selection | CASEWHERE expression evaluates to  choice a: process a  choice b: process b  …  OTHERWISE: default process  END CASE |
| Nested if | IF condition A THEN  process 1  ELSEIF condition B THEN  process 2  ELSEIF condition C THEN  process 3  ELSE  process 4  ENDIF |
| Repetition  Pre-test | WHILE condition is true  process  ENDWHILE |
| Repetition  Post-test | REPEAT  process  UNTIL condition is true |
| Repetition  For/Next | FOR variable = start TO finish STEP increment  statements  NEXT variable |

**Teacher note:** teachers can use a variety of websites to supplement information for this activity including [How to write pseudocode](https://www.indeed.com/career-advice/career-development/pseudocode).

**Activity 36:** construct a flowchart for pseudocode

Using knowledge of flowcharts, students are required to construct flowchart representations for the above pseudocode.

Construct your flowchart in the space below:

|  |
| --- |
|  |

**Teacher note:** the Stage 6 course specifications for Software Engineering demonstrate the comparison between pseudocode and flowcharts very clearly.

## Desk checking

As a class, walk through the below algorithms demonstrating how to conduct a desk check.

Figure 3 – algorithm 1 – simple addition – sequence

# Start
# Input A and B (two numbers)
A = int(input("Enter the first number (A): "))
B = int(input("Enter the first number (B): "))

# Calculate the sum of A and B (C = A + B)
C = A + B

#Output C
Print ("The sum of A and B is:", C)

#Stop


|  |
| --- |
| **Sample answer:**  A=3; B=5  A table that shows A= 3, B =5 and C=8 Therefore the output is the sum of A and B is 8. |

Figure 4 – algorithm 2 – finding the maximum of 3 numbers – selection (branching)

# Start
# Input three numbers: A, B, C 
A = int(input("Enter the first number (A): "))
B = int(input("Enter the first number (B): "))
C = int(input("Enter the first number (C): "))

# Check for the maximum
If A >= B and A>= C:
# Output A as the maximum
Print ("The maximum is A:", A)
Elif B >= A and B>= C:
# Output B as the maximum 
Print (The maximum is B:", B)
Else:
# Output C as the maximum 
Print (The maximum is C:", C)

# Stop

|  |
| --- |
| **Sample answer:**  A=7; B=12; C=5  A table that shows A= 7, B =12 and C=5 Therefore the maximum is: 12. |

Figure 5 – algorithm 3 – While loop to find the sum of N numbers – iteration (loop)

# Start
# Input N (a positive integer)
N = int(input("Enter a positive integer (N): "))

# Initialise variables
sum_of_numbers = 0
count = 1

# While loop to find the sum of the first N numbers
while count <= N:
sum_of_numbers += count
Count += 1

# Output the sum
print ("The sum of the first", N, "number is:", sum_of_numbers)

# Stop

|  |
| --- |
| **Sample answer:**  N=5; sum\_of\_numbers=0; count=1  A table that shows the sum of the first 5 numbers is 15. |

**Activity 37:** create a flowchart

Create a flowchart for the following algorithms:

1. Simple addition.
2. Finding the maximum of 3 numbers (branching).
3. While loop to find the sum of N numbers – iteration (loop).

Algorithm 1 – simple addition

|  |
| --- |
|  |

Algorithm 2 – finding the maximum of 3 numbers (branching)

|  |
| --- |
|  |

Algorithm 3 – while loop to find the sum of N numbers – iteration (loop)

|  |
| --- |
|  |

## Real-world problems that can be solved by an app

**Activity 38:** school canteen ordering

**Scenario**: your school does not have an online ordering system for their canteen. They still rely on cash payments and brown paper bag ordering. Parents, students and staff have expressed an interest in an app that could be used to order lunch and recess.

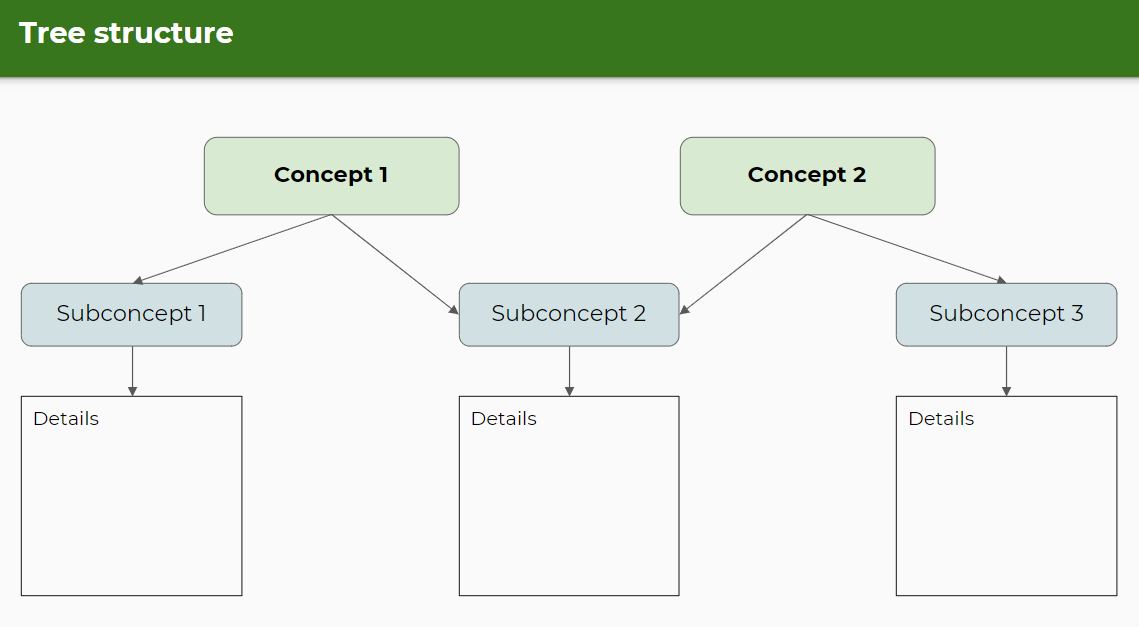
In pairs, you are going to complete the following tasks:

1. Evaluate some canteen ordering websites. Complete a [Plus, Minus, Interesting](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/551?clearCache=a604987f-48b9-6cdf-ed4a-ba6b278de07f) evaluation, of at least 2 ordering websites. Examples are listed below:
2. [QuickCliQ](https://www.quickcliq.com.au/)
3. [Flexischools](https://flexischools.com.au/schools)
4. [Munch Monitor](https://www.munchmonitor.com/)
5. [Spriggy Schools](https://www.spriggyschools.com.au/).
6. Consider positives, negatives and any interesting features that each site has to offer. You will find this information by looking at their frequently asked questions and parents and school tabs.
7. Design some interview questions to ask students, parents and staff about what kind of app they would like to use for online ordering. As a class watch [Designing a survey](https://www.youtube.com/watch?v=mdVWbuffdNY) (5:20) to help formulate your questions.
8. Once you have designed your questions and they have been approved by your teacher, design a form in Google or Microsoft that could be sent to participants.

**Teacher note**: the survey does not have to be distributed – the aim is to have students think about effective questioning. If the teacher wishes to collect and analyse data, a class survey could be developed.

1. Now, think about how the app would work. [Brainstorm](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/542) what you think the app would need to be successful, by breaking it down into parts – for example the canteen menu, ordering system, parent account and student account.
2. Map out a plan for your app using Slide 8 – [tree structure concept map](https://app.education.nsw.gov.au/digital-learning-selector/LearningActivity/Card/577) similar to the example shown below.

Figure 6 – tree structure concept map



**Teacher note:** this activity complements the [Oracle Digital Literacy program](https://education.oracle.com/odl). In this course students will learn the foundations of Oracle APEX, a low-code development platform that can be used to deliver learning on how to develop an app and guide students to build a café ordering system.

Low-code is a visual app development technology that uses pre-built modules and drag-and-drop editors to make app development easy.

The course is 20 hours and designed to be delivered across 28 periods and could be used in blending the delivery of Developing apps and web software with Designing for user experience.

## Error type in programming

**Syntax errors** are detected by the compiler or interpreter during code compilation and prevent the program from running.

**Logic errors** result from flawed program logic and lead to incorrect behaviour but don't cause program crashes.

**Runtime errors** occur during program execution and can crash the program or produce error messages that need to be handled.

**Activity 39:** research task – error type in programming

Complete Table 32 below with definitions, examples and the impact that error has on running code.

Table 32 – errors in running code and their impacts

|  |  |  |  |
| --- | --- | --- | --- |
| Errors | Description | Example | Impact |
| Syntax | Syntax errors, also known as compilation errors, occur when the code violates the rules of the programming language's syntax.  These errors prevent the program from being compiled or interpreted. | Missing semicolons or parentheses.  Using undeclared variables.  Incorrect capitalisation of function or variable names. | Syntax errors are detected by the compiler or interpreter during the compilation or parsing phase.  They prevent the program from being executed until fixed. Programs with syntax errors won't run at all. |
| Logic | Logic errors occur when the program's logic is flawed, resulting in unintended or incorrect behaviour.  These errors do not cause the program to crash but lead to unexpected results. | A mathematical calculation that produces the wrong result.  A conditional statement that doesn't correctly evaluate a condition.  A loop that doesn't terminate when it should. | Logic errors can be challenging to detect because the program still runs.  However, they can lead to incorrect output or behaviour and debugging is required to identify and fix them. |
| Runtime | Runtime errors, also known as exceptions or run-time exceptions, occur when the program is running and encounters a condition that it cannot handle.  These errors can lead to the program crashing or behaving unpredictably. | Division by zero.  Attempting to access an array element that doesn't exist.  Trying to open a file that doesn't exist. | Runtime errors can cause the program to terminate abruptly or produce error messages.  They need to be handled through exception handling mechanisms to prevent program crashes. |

**Activity 40:** find and correct the errors

Using the algorithms in Table 33 below, spot the error and provide a correction for the algorithm, including how the code will be changed.

Table 33 – coding errors in sample algorithms

|  |  |
| --- | --- |
| Sample algorithm | Spot the error |
| # Algorithm: Syntax Error Example  # Start # Input two numbers num1 = int(input("Enter the first number: ")) num2 = int(input("Enter the second number: "))  # Calculate their sum sum = num1 + num2  # Output the result print("The sum of", num1, "and", num2, "is:", sum) | The word sum is an in-built function, and therefore is a syntax error.  To fix this, a new variable name needs to be provided for the ’sum‘ – for example this could be changed to result. |
| # Algorithm: Logic Error Example  # Start # Input two numbers num1 = int(input("Enter the first number: ")) num2 = int(input("Enter the second number: "))  # Check if both numbers are positive if num 1 > 0 and num@> 0: Print ("Both numbers are positive.") else:  Print ("At lease one number is not positive.") # Stop | The if statement checks if the number is greater than 0 but doesn’t check if the number is 0 which is also a positive number.  To fix this, you need to add = to both conditions to ensure 0 is checked too. |
| # Algorithm: Runtime Error Example  # Start # Input two numbers num1 = int(input("Enter the first number: ")) num2 = int(input("Enter the second number: "))  # Attempt to divide num1 by num2 result = num1 / num2  # Output the result print ("The result of the division is:", result) # Stop | If num2 is entered as zero then this algorithm will result in a ZeroDivisionError, therefore a conditional statement if num2 is not equal to zero. |

**Teacher note:** depending on the software that is being used to create app solutions, students will need to be taught how to debug software using appropriate debugging techniques for example breakpoints, watch statements or single-line stepping.

# Producing and implementing

Most sections within this part of the syllabus should be completed through a practical project.

* Produce and implement an app using the preferred design in a general-purpose or object-oriented programming language.
* Develop the user interface (UI) and user experience (UX) of an app, including using event-driven programming to respond to user input.
* Develop a web page or app that separates content and presentation using html and cascading style sheet (CSS)
* Document the design and implementation of the solution in a project notebook.
* Interpret and modify existing programs (code) for apps.
* Design and implement modular programs (code) with functions for apps.
* Apply selected algorithms and data structures for apps.
* Validate programs using test cases and debug a range of errors.
* Interpret and extend or implement an object-oriented program (code)
* Select and use specialist terminology in context.
* Create a record of project development demonstrating iterative design and evaluation.

## Record of project development

Plan and manage a project using an iterative approach. Keep a record of project development and video record your system model attempting processes as it evolves.

Use the following pages as a diary to document the development of your project. Make note of your skills and knowledge gained, challenges faced and your successes demonstrating iterative design and evaluation.

Table 34 – project development diary

|  |  |
| --- | --- |
| Date | Description |
| \_\_/\_\_/\_\_ |  |
| \_\_/\_\_/\_\_ |  |
| \_\_/\_\_/\_\_ |  |
| \_\_/\_\_/\_\_ |  |
| \_\_/\_\_/\_\_ |  |
| \_\_/\_\_/\_\_ |  |

## Practical development

#### Option: Anvil – anvil.works

Login to [anvil.works](https://anvil.works/).

**Teacher note:** email the company and receive an educational licence. Students click on the provided link to gain a seat on your development team. This allows students full professional capabilities and the ability to add other developers on the licence as collaborators.

Anvil is a web-based programming option that focuses on Python with integrated HTML/CSS. There are over 30 tutorials on how to utilise the software to its full capabilities that range from beginner to advanced.

The following tutorials allow students to understand the functionality:

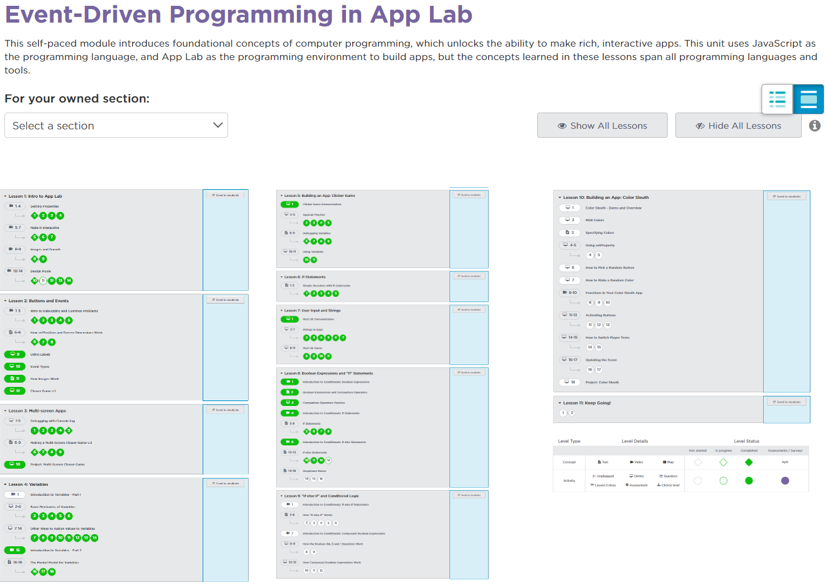
* [Simple feedback form](https://anvil.works/learn/tutorials/feedback-form)
* [Multi-user application](https://anvil.works/learn/tutorials/multi-user-apps)
* [Data grids: displaying data in tables](https://anvil.works/learn/tutorials/data-grids)
* [Database-backed apps](https://anvil.works/learn/tutorials/database-backed-apps)
* [To-do list app](https://anvil.works/learn/workshops/todo-list)

#### Option: App Lab – Code.org

Login to [Code.org](https://code.org/).

The Code Org App lab contains 11 self-paced lessons. Students can complete all activities to gain skills and knowledge to build their own apps. At any point students may start to develop their own projects. The screen captures below show the lessons and the progress of a learner. As each of the lessons are completed, they are marked in green. Teachers can check student progress and suggest changes or updates for students to consider. Programming can be completed using Javascript, however block coding is available to be used also as a check for errors. Your teacher will indicate where it is appropriate to use text-based programming. The module is self-paced. Do not rush to complete the activities.

Figure 7 – App Lab screenshots



# Testing and evaluating

These sections of the syllabus should be completed through a practical project.

* Evaluate their own project and that of their peers using predetermined functional and non-functional requirements.
* Validate algorithms and programs through tracing and test cases.
* Test and evaluate the functionality and performance of a simulation or game for specified requirements.

Students should conduct continuous evaluations throughout the Assessment task and peer-evaluation to gather feedback for improving the final solution.

Evidence of testing through screenshots and explanations of how they have fixed the errors should be displayed.

**Teacher note:** useful resources for evaluation include the [Teacher workbook from Grok Academy DT Applied](https://groklearning.com/course/aca-dt-78-py-assessment/). This resource walks through the whole design process from brainstorming to evaluation and pitching.

## Explore careers in software development

Choose a career in software development from the following list:

* Software Engineer
* Web Developer
* Mobile App Developer
* Data Scientist
* DevOps Engineer
* UX/UI Designer
* Quality Assurance Tester
* Database Administrator
* IT Project Manager
* Full Stack Developer.

**Activity 41:** explore careers in software development

Research one of the careers listed above using the criteria below:

* identify the education, skills and experience you may need for the chosen career. Including certifications and tertiary education (degrees) relevant to the chosen career
* research the availability of jobs in the chosen career using websites such as [SEEK](https://www.seek.com.au/), [Indeed](https://au.indeed.com/) and [CareerOne](https://www.careerone.com.au/). Include information such as wage range, working conditions and hours in your research
* investigate technical skills, programming languages and other additional information relating to the career
* Outline the roles and responsibilities a person may be required to perform when working in this role
* investigate various opportunities for professional development, such as online courses, coding bootcamps, conferences and industry certifications. Research and identify specific learning resources and opportunities that can enhance their skills and knowledge in their chosen career path.

Create an infographic to display your findings. Infographics can be created using [Canva](https://app.education.nsw.gov.au/digital-learning-selector/LearningTool/Card/653?clearCache=61f5576b-6b39-ccb3-cb77-3305363641d4).

**Teacher note:** the activity could be completed by individual students or in groups with a presentation component.

**Activity 42:** existing positions in software development

Research 3 different existing positions in software development and their criteria from current employment websites such as [SEEK](https://www.seek.com.au/), [Indeed](https://au.indeed.com/) and [CareerOne](https://www.careerone.com.au/).

Table 35 – position 1

|  |  |
| --- | --- |
| Criteria | Response |
| Job title |  |
| Training required |  |
| Personal requirements |  |
| Outline of duties |  |
| Average income |  |
| Working hours |  |

Table 36 – position 2

|  |  |
| --- | --- |
| Criteria | Response |
| Job title |  |
| Training required |  |
| Personal requirements |  |
| Outline of duties |  |
| Average income |  |
| Working hours |  |

Table 37 – position 3

|  |  |
| --- | --- |
| Criteria | Response |
| Job title |  |
| Training required |  |
| Personal requirements |  |
| Outline of duties |  |
| Average income |  |
| Working hours |  |

# References

This resource contains NSW Curriculum and syllabus content. The NSW Curriculum is developed by the NSW Education Standards Authority. This content is prepared by NESA for and on behalf of the Crown in right of the State of New South Wales. The material is protected by Crown copyright.

Please refer to the NESA Copyright Disclaimer for more information <https://educationstandards.nsw.edu.au/wps/portal/nesa/mini-footer/copyright>.

NESA holds the only official and up-to-date versions of the NSW Curriculum and syllabus documents. Please visit the NSW Education Standards Authority (NESA) website <https://educationstandards.nsw.edu.au> and the NSW Curriculum website <https://curriculum.nsw.edu.au>.

[Computing Technology 7–10 Syllabus](https://curriculum.nsw.edu.au/syllabuses/computing-technology-7-10-2022) © NSW Education Standards Authority (NESA) for and on behalf of the Crown in the State of New South Wales, 2022.

accessiBe Team (10 December 2023) ‘[Slide into Inclusion with Accessible Presentation Decks](https://accessibe.com/blog/knowledgebase/accessible-presentation-decks)’, accessiBe blog, accessed 17 September 2024.

AltexSoft (30 November 2023) ‘[Functional and Nonfunctional Requirements: Specification and Types](https://www.altexsoft.com/blog/functional-and-non-functional-requirements-specification-and-types/#:~:text=Functional%20requirements%C2%A0define,quality%20attributes.)’, AltexSoft blog, accessed 17 September 2024.

AltexSoft (16 June 2022) ['What are Non-functional Requirements and How Do They Work?’ [video]](https://www.youtube.com/watch?v=fc-5HJPBZMQ), AltexSoft, YouTube, accessed 17 September 2024.

Anvil (2024a) [*Anvil Works*](https://anvil.works/) [website], accessed 17 September 2024.

—— (2024b) ‘[Build a Simple Feedback Form](https://anvil.works/learn/tutorials/feedback-form)’, Tutorials, Anvil website, accessed 17 September 2024.

—— (2024c) ‘[Build Database-Backed Apps](https://anvil.works/learn/tutorials/database-backed-apps)’, Tutorials, Anvil website, accessed 17 September 2024.

—— (2024d) ‘[Data Grids: Displaying Data in Tables](https://anvil.works/learn/tutorials/data-grids)’, Tutorials, Anvil website, accessed 17 September 2024.

—— (2024e) ‘[Multi-User Applications with Anvil](https://anvil.works/learn/tutorials/multi-user-apps)’, Tutorials, Anvil website, accessed 17 September 2024.

Arkenea, Inc (17 March 2016) ‘[The Evolution of Mobile Apps – 1994 through 2016](https://arkenea.com/blog/evolution-of-mobile-apps/)’, Arkenea blog, accessed 17 September 2024.

Attorney-General’s Department (n.d.) ‘[Copyright basics: What does copyright protect?](https://www.ag.gov.au/rights-and-protections/copyright/copyright-basics#:~:text=Copyright%20provides%20legal%20protection%20for,expression%20of%20ideas%20or%20information.)’, Copyright, Attorney-General's Department website, accessed 17 September 2024.

Canva (n.d.) [*Canva*](http://www.canva.com/) [website], accessed 17 September 2024.

CareerOne Pty Ltd (2024) [*CareerOne*](https://www.careerone.com.au/) [website], accessed 17 September 2024.

CareerFoundry (30 October 2018) [How To Create Your First Wireframe (Video Guide)](https://www.youtube.com/watch?v=KdfO_e0yK-g) *CareerFoundry* YouTube, accessed 17 September 2024.

Code.org (2024) [*Code.org*](https://code.org/), accessed 17 September 2024.

DLTV (Digital Learning and Teaching Victoria) (2022) [*Object-Oriented Programming: getting started for Australian teachers*](https://sites.google.com/dltv.vic.edu.au/object-oriented), DLTV website, accessed 17 September 2024.

Eye on Tech (16 September 2020) ['What is a Web App? Web App vs. Native App' [video]](https://www.youtube.com/watch?v=qt6gSW-uYKI), *Eye on Tech*, YouTube, accessed 17 September 2024.

Figma (n.d.) ‘[What is wireframing?](https://www.figma.com/resource-library/what-is-wireframing/)’, Design basics, Figma website, accessed 17 September 2024.

Flexischools (n.d.) [*Flexischools: Online canteen ordering system for schools*](https://flexischools.com.au/schools), Flexischools website, accessed 17 September 2024.

Grok Academy Limited (n.d.) [*Grok Academy*](https://groklearning.com/) [website], accessed 17 September 2024.

Indeed (2024) [*Indeed Australia*](https://au.indeed.com/) [website], accessed 17 September 2024.

Inventionland (2024) [‘The History of Mobile Apps](https://inventionland.com/blog/the-history-of-mobile-apps/)’, Inventionland blog, accessed 17 September 2024.

IP (Intellectual Property) Australia (n.d.) [*Understanding intellectual property (IP)*](https://www.ipaustralia.gov.au/understanding-ip), IP Australia website, accessed 17 September 2024.

Jabrils (9 July 2019) [’What are Data Types?’ [video]](https://www.youtube.com/watch?v=A37-3lflh8I), Jabrils, YouTube, accessed 17 September 2024.

LearnFree (4 October 2018) [’Computer Science Basics: Sequences, Selections, and Loops’ [video]](https://www.youtube.com/watch?v=eSYeHlwDCNA), LearnFree, YouTube, accessed 17 September 2024.

LinkedIn (2024) ‘[Reverse Engineering Mobile Apps: Analysis and Documentation](https://www.linkedin.com/advice/3/how-do-you-analyze-document-results-reverse#:~:text=Reverse%20engineering%20mobile%20apps%20is,apps%20based%20on%20existing%20ones.)’, Learn more about Collaborative Articles, LinkedIn website, accessed 17 September 2024.

Lucid Software Inc (2024) [*Flowchart Symbols and Notation*](https://www.lucidchart.com/pages/flowchart-symbols-meaning-explained), Lucidchart website, accessed 17 September 2024.

Motiso D (2024) ‘[How to write pseudocode (Definition, Components and Pros)](https://www.indeed.com/career-advice/career-development/pseudocode)’, Career development, Indeed website, accessed 17 September 2024.

Mr Matthews (5 March 2018) [‘CS Basics: Input Process Output’ [video]](https://www.youtube.com/watch?v=CBf-jIn44X0), Mr Matthews, YouTube, accessed 17 September 2024.

MunchMonitor (2021) [*MunchMonitor*](https://www.munchmonitor.com/) [website], accessed 17 September 2024.

Oracle (n.d.) [*Oracle Digital Literacy*](https://education.oracle.com/odl), Oracle University website, accessed 17 September 2024.

Perforce Software Inc (15 November 2019) ‘[Guide to Flowchart Symbols, from Basic to Advanced](https://www.gliffy.com/blog/guide-to-flowchart-symbols)’, Gliffy by Perforce blog, accessed 17 September 2024.

QuickCliq (2024) [*QuickCliq*](https://www.quickcliq.com.au/) [website], accessed 17 September 2024.

Refsnes Data (1999–2024) [*Python Tutorial*](https://www.w3schools.com/python/), W3Schools website, accessed 17 September 2024.

Scott T (4 February 2020) [‘Why TRUE + TRUE = 2: Data Types’ [video]](https://www.youtube.com/watch?v=6otW6OXjR8c), Tom Scott, YouTube, accessed 17 September 2024.

SEEK (n.d.) [*SEEK Australia*](https://www.seek.com.au/) [website], accessed 17 September 2024.

Simplilearn (11 June 2020) ['What Is Cyber Security | How It Works? | Cyber Security In 7 Minutes | Cyber Security | Simplilearn' [video]](https://www.youtube.com/watch?v=inWWhr5tnEA), Simplilearn, YouTube, accessed 17 September 2024.

Spriggy Schools (2023) [*Spriggy Schools*](https://www.spriggyschools.com.au/) [website], accessed 17 September 2024.

SVB Enterprises Pty Ltd trading as PitchVest (6 February 2020) ‘[What is a Pitch Deck?](https://www.pitchvest.com.au/post/what-is-a-pitch-deck)’, PitchVest blog, accessed 17 September 2024.

Taylor S (2024) ‘[To-do List App](https://anvil.works/learn/workshops/todo-list)’, Workshops, Anvil website, accessed 17 September 2024.

TED (22 September 2023) [‘The Secret to Successfully Pitching an Idea | The Way We Work, a TED series’ [video]](https://www.youtube.com/watch?v=l0hVIH3EnlQ), TED, YouTube, accessed 17 September 2024.

Tutorials Point (2024) ‘[Flowchart Elements](https://www.tutorialspoint.com/programming_methodologies/programming_methodologies_flowchart_elements.htm)’, Programming Methodologies Tutorial, Tutorials Point website, accessed 17 September 2024.

Veselski A (2 June 2022) ‘[20+ mobile app wireframe examples to inspire you](https://decode.agency/article/mobile-app-wireframe-examples/)’, The DECODE blog, accessed 17 September 2024.

Zen Flowchart (n.d.) [*Flowchart Symbols – A Complete Guide*](https://www.zenflowchart.com/flowchart-symbols), Zen Flowchart website, accessed 17 September 2024.

**© State of New South Wales (Department of Education), 2024**

The copyright material published in this resource is subject to the Copyright Act 1968 (Cth) and is owned by the NSW Department of Education or, where indicated, by a party other than the NSW Department of Education (third-party material).

Copyright material available in this resource and owned by the NSW Department of Education is licensed under a [Creative Commons Attribution 4.0 International (CC BY 4.0) license](https://creativecommons.org/licenses/by/4.0/).

[](https://creativecommons.org/licenses/by/4.0/)

This license allows you to share and adapt the material for any purpose, even commercially.

Attribution should be given to © State of New South Wales (Department of Education), 2024.

Material in this resource not available under a Creative Commons license:

* the NSW Department of Education logo, other logos and trademark-protected material
* material owned by a third party that has been reproduced with permission. You will need to obtain permission from the third party to reuse its material.

**Links to third-party material and websites**

Please note that the provided (reading/viewing material/list/links/texts) are a suggestion only and implies no endorsement, by the New South Wales Department of Education, of any author, publisher, or book title. School principals and teachers are best placed to assess the suitability of resources that would complement the curriculum and reflect the needs and interests of their students.

If you use the links provided in this document to access a third-party's website, you acknowledge that the terms of use, including licence terms set out on the third-party's website apply to the use which may be made of the materials on that third-party website or where permitted by the Copyright Act 1968 (Cth). The department accepts no responsibility for content on third-party websites.