Digital Solutions 2025 v1.0

General senior syllabus

January 2024

Queensland Government

Queensland Curriculum & Assessment Authority

For all Queensland schools

ISBN

Electronic version: 978-1-74378-290-3

Queensland Curriculum & Assessment Authority
PO Box 307 Spring Hill QLD 4004 Australia

Phone: (07) 3864 0299
Email: office@qcaa.qld.edu.au
Website: www.qcaa.qld.edu.au

Contents

Queensland syllabuses for senior subjects

In Queensland, a syllabus for a senior subject is an official 'map' of a senior school subject. A syllabus's function is to support schools in delivering the Queensland Certificate of Education (QCE) system through high-quality and high-equity curriculum and assessment.

Syllabuses are based on design principles developed from independent international research about how excellence and equity are promoted in the documents teachers use to develop and enliven the curriculum.

Syllabuses for senior subjects build on student learning in the Prep to Year 10 Australian Curriculum and include General, General (Extension), Senior External Examination (SEE), Applied, Applied (Essential) and Short Course syllabuses.

More information about syllabuses for senior subjects is available at www.qcaa.qld.edu.au/senior/senior-subjects and in the 'Queensland curriculum' section of the *QCE and QCIA policy and procedures handbook*.

Teaching, learning and assessment resources will support the implementation of a syllabus for a senior subject. More information about professional resources for senior syllabuses is available on the QCAA website and via the QCAA Portal.

Course overview

Rationale

Technologies have been an integral part of society for as long as humans have had the desire to create solutions to improve their own and others' quality of life. Technologies have an impact on people and societies by transforming, restoring and sustaining the world in which we live.

Australia needs enterprising and innovative individuals with the ability to make discerning decisions concerning the development, use and impact of technologies. When developing technologies, these individuals need to be able to work independently and collaboratively to solve open-ended problems. Subjects in the Technologies learning area prepare students to be effective problem-solvers as they learn about and work with contemporary and emerging technologies.

In Digital Solutions, students learn about algorithms, computer languages and user interfaces through generating digital solutions to problems. They engage with data, information and applications to generate digital solutions that filter and present data in timely and efficient ways while understanding the need to encrypt and protect data. They understand computing's personal, social and economic impact, and the issues associated with the ethical integration of technology into our daily lives.

Students engage in problem-based learning that enables them to explore and develop ideas, generate digital solutions, and evaluate impacts, components and solutions. They understand that solutions enhance their world and benefit society. To generate digital solutions, students analyse problems and apply computational, design and systems thinking processes. Students understand that progress in the development of digital solutions is driven by people and their needs.

Learning in Digital Solutions provides students with opportunities to develop, generate and repurpose solutions that are relevant in a world where data and digital realms are transforming entertainment, education, business, manufacturing and many other industries. Australia's workforce and economy requires people who are able to collaborate, use creativity to be innovative and entrepreneurial, and transform traditional approaches in exciting new ways.

By using the problem-based learning framework, students develop confidence in dealing with complexity, as well as tolerance for ambiguity and persistence in working with difficult problems that may have many solutions. Students are able to communicate and work with others in order to achieve a common goal or solution. Students write computer programs to generate digital solutions that use data; require interactions with users and within systems; and affect people, the economy and environments. Solutions are generated using combinations of readily available hardware and software development environments, code libraries or specific instructions provided through programming. Some examples of digital solutions include instructions for a robotic system, an instructional game, a productivity application, products featuring interactive data, animations and websites.

Digital Solutions prepares students for a range of careers in a variety of digital contexts. It develops thinking skills that are relevant for digital and non-digital real-world challenges. It prepares them to be successful in a wide range of careers and provides them with skills to engage in and improve the society in which we work and play. Digital Solutions develops the 21st century skills of critical and creative thinking, communication, collaboration and teamwork, personal and social skills, and information and communication technologies (ICT) skills that are critical to students' success in further education and life.

Syllabus objectives

The syllabus objectives outline what students have the opportunity to learn.

1. **Recognise and describe elements, components, principles and processes.**

   When students recognise, they identify or recall facts and particular features of elements, components, principles and processes used in digital technologies. When students describe, they give an account of elements, components, principles and processes in technology contexts.

2. **Symbolise and explain information, ideas and interrelationships.**

   When students symbolise, they represent information, idea development and system interrelationships using models, sketches, diagrams, tables and/or schemas. When students explain, they make information, ideas and interrelationships clear by describing them in more detail or revealing relevant facts.

3. **Analyse problems and information.**

   When students analyse, they breakdown and examine problems and information to ascertain patterns, similarities and differences in order to identify elements, components and features, and their relationship to the structure of problems. They determine the logic and reasonableness of information by using systems thinking and decomposition, pattern recognition, and abstraction computational thinking.

4. **Determine solution requirements and criteria.**

   When students determine solution requirements and success criteria, they establish, conclude or ascertain the interface, algorithm, programming and identified solution needs and constraints.

5. **Synthesise information and ideas to determine possible digital solutions.**

   When students synthesise, they combine and integrate information and ideas, and resolve uncertainties using design, systems and computational thinking to create new understanding and identify possible digital solutions.

6. **Generate components of the digital solution.**

   When students generate, they use information, software, programming tools and skills to create components of an identified digital solution.

7. **Evaluate impacts, components and solutions against criteria to make refinements and justified recommendations.**

   When students evaluate, they appraise impacts, components and solutions by weighing up or assessing strengths, implications and limitations against success criteria. When students make refinements, they make partial or minor changes to improve the user experience and technical operation based on criteria. They use testing to evaluate and refine components and solutions based on criteria. When students make justified recommendations, they use supporting evidence to suggest modifications or enhancements.

8. **Make decisions about and use mode-appropriate features, language and conventions for particular purposes and contexts.**

   When students make decisions about mode-appropriate features, language and conventions, they use written, visual and spoken features to express meaning for particular purposes in a range of contexts. Written communication includes language conventions, specific vocabulary and language features such as annotations, paragraphs and sentences. Visual communication includes photographs, sketches, drawings, diagrams and motion graphics. Visual features include the elements and principles of visual communication. Spoken communication includes verbal and nonverbal features and may be for live or virtual audiences. Students use referencing conventions to practise ethical scholarship.

Designing a course of study in Digital Solutions

Syllabuses are designed for teachers to make professional decisions to tailor curriculum and assessment design and delivery to suit their school context and the goals, aspirations and abilities of their students within the parameters of Queensland's senior phase of learning.

The syllabus is used by teachers to develop curriculum for their school context. The term *course of study* describes the unique curriculum and assessment that students engage with in each school context. A course of study is the product of a series of decisions made by a school to select, organise and contextualise subject matter, integrate complementary and important learning, and create assessment tasks in accordance with syllabus specifications.

It is encouraged that, where possible, a course of study is designed such that teaching, learning and assessment activities are integrated and enlivened in an authentic setting.

Course structure

Digital Solutions is a General senior syllabus. It contains four QCAA-developed units from which schools develop their course of study.

Each unit has been developed with a notional time of 55 hours of teaching and learning, including assessment.

Students should complete Unit 1 and Unit 2 before beginning Units 3 and 4. Units 3 and 4 are studied as a pair.

More information about the requirements for administering senior syllabuses is available in the 'Queensland curriculum' section of the *QCE and QCIA policy and procedures handbook*.

Curriculum

Senior syllabuses set out only what is essential while being flexible so teachers can make curriculum decisions to suit their students, school context, resources and expertise.

Within the requirements set out in this syllabus and the *QCE and QCIA policy and procedures handbook*, schools have autonomy to decide:

- how and when subject matter is delivered
- how, when and why learning experiences are developed, and the context in which learning occurs
- how opportunities are provided in the course of study for explicit and integrated teaching and learning of complementary skills.

These decisions allow teachers to develop a course of study that is rich, engaging and relevant for their students.

Assessment

Senior syllabuses set out only what is essential while being flexible so teachers can make assessment decisions to suit their students, school context, resources and expertise.

General senior syllabuses contain assessment specifications and conditions for the assessment instruments that must be implemented with Units 3 and 4. These specifications and conditions ensure comparability, equity and validity in assessment.

Within the requirements set out in this syllabus and the *QCE and QCIA policy and procedures handbook*, schools have autonomy to decide:

- specific assessment task details
- assessment contexts to suit available resources
- how the assessment task will be integrated with teaching and learning activities
- how authentic the task will be.

In Unit 1 and Unit 2, schools:

- develop at least two but no more than four assessments
- complete at least one assessment for each unit
- ensure that each unit objective is assessed at least once.

In Units 3 and 4, schools develop three assessments using the assessment specifications and conditions provided in the syllabus.

More information about assessment in senior syllabuses is available in ‘The assessment system’ section of the *QCE and QCIA policy and procedures handbook*.

Subject matter

Each unit contains a unit description, unit objectives and subject matter. Subject matter is the body of information, mental procedures and psychomotor procedures (see Marzano & Kendall 2007, 2008) that are necessary for students’ learning and engagement with the subject. Subject matter itself is not the specification of learning experiences but provides the basis for the design of student learning experiences.

Subject matter has a direct relationship with the unit objectives and provides statements of learning that have been constructed in a similar way to objectives.

Aboriginal perspectives and Torres Strait Islander perspectives

The QCAA is committed to reconciliation. As part of its commitment, the QCAA affirms that:

- Aboriginal peoples and Torres Strait Islander peoples are the first Australians, and have the oldest living cultures in human history
- Aboriginal peoples and Torres Strait Islander peoples have strong cultural traditions and speak diverse languages and dialects, other than Standard Australian English
- teaching and learning in Queensland schools should provide opportunities for students to deepen their knowledge of Australia by engaging with the perspectives of Aboriginal peoples and Torres Strait Islander peoples
- positive outcomes for Aboriginal students and Torres Strait Islander students are supported by successfully embedding Aboriginal perspectives and Torres Strait Islander perspectives across planning, teaching and assessing student achievement.

Guidelines about Aboriginal perspectives and Torres Strait Islander perspectives and resources for teaching are available at www.qcaa.qld.edu.au/k-12-policies/aboriginal-torres-strait-islander-perspectives.

Where appropriate, Aboriginal perspectives and Torres Strait Islander perspectives have been embedded in the subject matter.

Complementary skills

Opportunities for the development of complementary skills have been embedded throughout subject matter. These skills, which overlap and interact with syllabus subject matter, are derived from current education, industry and community expectations and encompass the knowledge, skills, capabilities, behaviours and dispositions that will help students live and work successfully in the 21st century.

These complementary skills are:

- literacy — the knowledge, skills, behaviours and dispositions about language and texts essential for understanding and conveying English language content
- numeracy — the knowledge, skills, behaviours and dispositions that students need to use mathematics in a wide range of situations, to recognise and understand the role of mathematics in the world, and to develop the dispositions and capacities to use mathematical knowledge and skills purposefully
- 21st century skills — the attributes and skills students need to prepare them for higher education, work, and engagement in a complex and rapidly changing world. These skills include critical thinking, creative thinking, communication, collaboration and teamwork, personal and social skills, and digital literacy. The explanations of associated skills are available at www.qcaa.qld.edu.au/senior/senior-subjects/general-subjects/21st-century-skills.

It is expected that aspects of literacy, numeracy and 21st century skills will be developed by engaging in the learning outlined in this syllabus. Teachers may choose to create additional explicit and intentional opportunities for the development of these skills as they design the course of study.

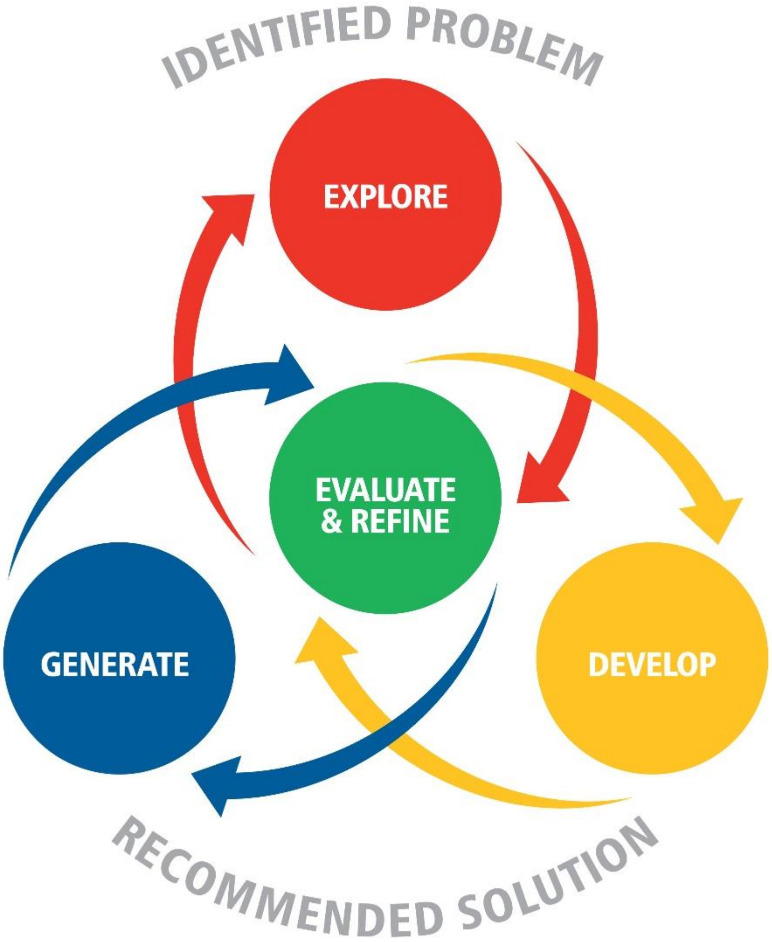
Additional subject-specific information

Additional subject-specific information has been included to support and inform the development of a course of study.

Procedural knowledge

The problem-solving process in Digital Solutions is analytical and technical in nature. The process is iterative and involves several phases. Students are required to explore problems, develop ideas, generate components and digital solutions. They evaluate components and the personal, social and economic impacts of their solutions to make refinements and recommendations.

Figure 1: Problem-solving process in Digital Solutions



Explore

The explore phase involves students investigating a need, want or opportunity to analyse and understand a digital problem and its relationship to existing solutions.

To explore the problem, students:

- describe problems from a user perspective
- recognise constraints
- use decomposition to dissect problems and existing solutions to similar problems
- use abstraction and pattern recognition to identify essential features of elements, components, relationships and structures of problems
- analyse problems, solutions and information to make decisions about the reasonableness of information and the structure, availability and accuracy of existing problems and solutions
- use systems thinking to identify and understand the relationships between users, solutions and the components of solutions in similar problems
- identify and understand possible solution requirements, such as information, skills and tools, by considering elements, components and features, and their relationship to the structure of the problem
- use design, systems and computational (decomposition, pattern recognition and abstraction) thinking processes to determine success criteria that are used to appraise and make decisions throughout, and at the end of, the problem-solving process in Digital Solutions. Success criteria are used to evaluate the personal, social and economic impacts, and quality, appropriateness and effectiveness of the generated component or solution
- use design thinking to evaluate ideas that best meet the success criteria.

Develop

The develop phase involves students creating new understanding and identifying possible solutions using design, systems, and abstraction and algorithmic computational thinking processes. Students evaluate personal, social and economic impacts, components and digital solutions against criteria throughout the develop phase to make decisions and refine the user experience and technical operation of components of the solution.

To develop ideas, students:

- use design thinking to visualise ideas and synthesise information and ideas in response to a digital problem by using drawing and creative skills to represent and communicate ideas
- acquire required information, tools and skills to implement a solution plan
- use computational thinking to apply abstraction procedures to problem components
- use computational thinking to express algorithms
- use systems and design thinking to develop ideas about components and solutions to test conceptual models
- use systems and design thinking to develop new ideas, identify a solution and evaluate ideas that best meet the criteria for success.

Generate

The generate phase involves students using information, software, programming tools and skills, and systems and design thinking processes to create components of an identified digital solution. Students evaluate personal, social and economic impacts, components and digital solutions against criteria throughout the generate phase to make decisions and refine the user experience and technical operation of components of the solution.

To generate solutions, students:

- use design and systems thinking processes to synthesise acquired information, ideas and skills to
  - generate individual components of a preferred solution
  - generate and refine a preferred solution in response to new or existing information
- use design thinking to evaluate and respond to the results of alpha testing
- use systems and design thinking to construct a solution and communicate knowledge and understanding of the solution.

Evaluate and refine

When students evaluate, they use systems, design and computational thinking to appraise personal, social and economic impacts, components and digital solutions by weighing up or assessing strengths, implications and limitations against success criteria. When students refine ideas and a digital solution, they make changes based on selected criteria to improve the user experience and technical operation. Evaluation occurs throughout each phase of the problem-solving process to refine the components and a solution in response to the success criteria.

To evaluate and refine, students:

- use pattern recognition to compare behaviours, e.g. usage and system, and outcomes of alternative solutions
- appraise test data and errors
- use design thinking to evaluate components and the digital solution against success criteria
- make changes in response to continual testing and appraisal of components and digital solutions
- make justified recommendations about inputs and the digital solution with supporting evidence.

Pseudocode

Pseudocode will be used as the formal method of describing algorithms in this syllabus. It is a descriptive method used to represent an algorithm and is a mixture of everyday language and programming conventions. Pseudocode is often an intermediate step in programming between planning and writing executable code.

Pseudocode does not have a standard format and varies between programmers; however, algorithms must be able to be understood by anyone independent of a particular programming language. When students use pseudocode, they should:

- implement the basic control structures of assignment, sequence, selection, condition, iteration and modularisation using capitalised keywords associated with the constructs
- indent algorithmic steps where appropriate
- consider a case style that works best with the programming language they will be using in their project, e.g. Pascal Case or Camel Case.

The following pseudocode demonstrates examples of assignment (`DECLARE`), sequence, condition (`IF`), selection (`THEN`), iteration (`WHILE`), modularisation (`FUNCTION, CALL`), indentation and two variations of case (`UserLogin, userName`) for a user authentication process:

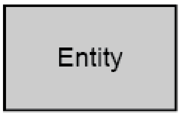
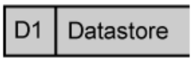
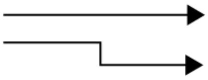
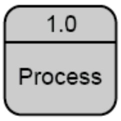
```
FUNCTION UserLogin():
    DECLARE userName, userPassword AS STRING
    DECLARE loginAttempts AS INTEGER = 3

    OUTPUT "Welcome to Login System."

    WHILE loginAttempts > 0
        INPUT "Enter username:" -> userName
        INPUT "Enter password:" -> userPassword

        IF IsValid(userName, userPassword) THEN
            OUTPUT "Login successful!"
            ShowDashboard()
            RETURN
        END IF

        loginAttempts -= 1
        OUTPUT "Invalid. " & loginAttempts & " attempts left."
    END WHILE
    OUTPUT "Try again later."
END FUNCTION

FUNCTION IsValid(user, pass) AS BOOLEAN:
    RETURN user = "admin" AND pass = "password123"
END FUNCTION

FUNCTION ShowDashboard():
    OUTPUT "Welcome to Dashboard."
END FUNCTION

CALL UserLogin()
```

Data processing system analysis and development

Data flow diagrams (DFD) — which include data source, data flow, data storage and process — are used to represent system interrelationships, data, system or process-oriented workflow.

Digital Solutions uses Gane-Sarson (1979) notation for data flow diagrams. Four basic symbols are used to visually represent data source, flow, storage and processes, as shown in Figure 2.

Figure 2: Data flow diagram symbols, names and functions

Symbol	Name	Function
	**Data source** or **External entity**	A source or destination of data flow that is outside the area of study
	**Data store**	Repository of data
	**Data flow**	A connector shows relationships between the representative shapes e.g. request/reply
	**Process**	Transforms incoming data flow into outgoing data flow

Conventions

When students develop data flow diagrams, they should:

- choose meaningful names for processes, flows and data stores, including
  - nouns for entities, data flows and data stores
  - verbs or verb-object names for processes that describes what the process does, e.g. check credentials
- number processes and data stores
- identify data stores as
  - D — data on disk (retained even if the system is turned off)
  - M — manual data (physical files, paper records or manually processed digital data)
  - T — temporary (not retained long-term) or transient (rapidly changing, e.g. RAM, cache).

Information comes from and goes to entities and data stores via processes, therefore entities may not:

- send data directly to other entities
- send data directly to data stores (data must be processed first)
- get data directly from data stores (data must be processed first).

Reporting

General information about determining and reporting results for senior syllabuses is provided in the ‘Determining and reporting results’ section of the *QCE and QCIA policy and procedures handbook*.

Reporting standards

Reporting standards are summary statements that describe typical performance at each of the five levels (A–E).

A
The student, in a range of digital technologies contexts, demonstrates discerning recognition and description of elements, components, principles and processes; adept symbolisation and effective explanation of relevant information, ideas and interrelationships. The student demonstrates insightful analysis of problems and contextual information, astute determination of solution requirements and success criteria against which to evaluate. The student demonstrates logical synthesis of information and ideas to determine possible digital solutions; sophisticated generation of components and digital solutions; critical evaluation of components and digital solutions against criteria and critical evaluation of impacts, with effective refinement and justification of recommendations; effective decision-making about, and fluent use of, mode-appropriate features, language and conventions for particular purposes and contexts.
B
The student, in a range of digital technologies contexts, demonstrates effective recognition and description of elements, components, principles and processes; methodical symbolisation and effective explanation of relevant information, ideas and interrelationships. The student demonstrates considered analysis of problems and relevant information, logical determination of solution requirements and success criteria against which to evaluate. The student demonstrates logical synthesis of relevant information and ideas to determine possible digital solutions; effective generation of components and digital solutions; reasoned evaluation of components and digital solutions against criteria and reasoned evaluation of impacts, with effective refinement and justification of recommendations; effective decision-making about, and fluent use of, mode-appropriate features, language and conventions for particular purposes and contexts.
C
The student, in a range of digital technologies contexts, demonstrates adequate recognition and description of elements, components, principles and processes; competent symbolisation and appropriate explanation of information, ideas and interrelationships. The student demonstrates adequate analysis of problems and information, reasonable determination of solution requirements and success criteria against which to evaluate. The student demonstrates adequate synthesis of information and ideas to determine possible digital solutions; adequate generation of components and digital solutions; feasible evaluation of components and digital solutions against criteria and feasible evaluation of impacts, with adequate refinement and justification of recommendations; appropriate decision-making about, and fluent use of, mode-appropriate features, language and conventions for particular purposes and contexts.

D
The student, in a range of digital technologies contexts, demonstrates making statements about elements, components, principles or processes; incomplete symbolisation and superficial explanation of information, ideas or interrelationships. The student demonstrates superficial analysis of problems or information, vague determination of solution requirements and success criteria against which to evaluate. The student demonstrates simple synthesis of information or ideas to determine possible digital solutions; basic generation of components and digital solutions; superficial evaluation of components or digital solutions against criteria or superficial evaluation of impacts; inadequate decision-making about, and inconsistent use of, mode-appropriate features, language and conventions for particular purposes and contexts.
E
The student, in a range of digital technologies contexts, demonstrates recognition of aspects of elements, components, principles or processes; limited symbolisation or explanation of information, ideas or interrelationships. The student demonstrates the making of statements about problems, information or solution requirements. The student demonstrates unclear combination of information or ideas about digital solutions; identification of a change to an idea or a solution; generation of elements of solution components; inadequate decision-making about, and inconsistent use of, mode-appropriate features, language and conventions for particular purposes and contexts.

Determining and reporting results

Unit 1 and Unit 2

Schools make judgments on individual assessment instruments using a method determined by the school. They may use the reporting standards or develop an instrument-specific marking guide (ISMG). Marks are not required for determining a unit result for reporting to the QCAA.

The unit assessment program comprises the assessment instrument/s designed by the school to allow the students to demonstrate the unit objectives. The unit judgment of A–E is made using reporting standards.

Schools report student results for Unit 1 and Unit 2 to the QCAA as satisfactory (S) or unsatisfactory (U). Where appropriate, schools may also report a not rated (NR).

Units 3 and 4

Schools mark each of the three internal assessment instruments implemented in Units 3 and 4 using ISMGs.

Schools report a provisional mark by criterion to the QCAA for each internal assessment.

Once confirmed by the QCAA, these results will be combined with the result of the external assessment developed and marked by the QCAA.

The QCAA uses these results to determine each student’s subject result as a mark out of 100 and as an A–E.

Units

Unit 1: Creating with code

In Unit 1, students will explore the creative and technical aspects of developing interactive digital solutions. They investigate algorithms, programming features and useability principles to generate small interactive solutions using programming tools and gain a practical understanding of programming features. This allows them the opportunity to explore existing and developing trends involving digital technologies.

Unit objectives

1. Recognise and describe programming features and useability principles.
2. Symbolise and explain information, ideas and interrelationships related to digital problems.
3. Analyse problems and information related to a selected technology context.
4. Determine user experience and programming requirements, and success criteria of a digital problem.
5. Synthesise information and ideas to determine possible prototype digital solutions.
6. Generate user interface and programmed components of the prototype digital solution.
7. Evaluate impacts, components and solutions against criteria to make refinements and justified recommendations.
8. Make decisions about and use mode-appropriate features, language and conventions for particular purposes and contexts.

Technology contexts

Schools select a single technology context to examine problems in this unit. Schools may select more than one programming language to cover the required operations to be performed. Selection and suitability of programming languages will vary depending on the school and technology context selected. Students must address both the subject matter and the programming features of the programming language in the selected technology context.

Technology context	Example languages
Web applications	JavaScript, Python, Java, Ruby, PHP, C#, Go
Mobile applications	Java, Swift, Kotlin, Dart, JavaScript, C#
Interactive media	JavaScript, C#, C++, Python
Intelligent systems	Python, R, Java, C++, Julia, Robot C

Subject matter

Topic 1: Understanding digital problems

- Understand methods of breaking down problems into parts using computational thinking and thinking tools, e.g. mind maps.
- Understand and describe personal, social and economic impacts.
- Analyse problems to identify
  - the human need, want or opportunity that requires a new or re-imagined digital solution
  - essential elements, components and features of problems in Digital Solutions
  - where and how digital technologies are used to solve problems to meet personal, societal and organisational needs, e.g. through search engines, robotics, mobile phone applications, automobile control systems, wearable devices, and the use of smart objects in the Internet of Things.
- Explore existing solutions to similar problems, e.g. existing applications, games or websites.
- Explore emerging technologies in relation to problems, e.g. machine learning, deep learning, neural networks, natural language processing.
- Analyse a given problem to identify
  - the scope of the problem
  - constraints and limitations of the environment
  - the requirements of the solution
  - the user perspective and user-experience requirements
  - technical issues of the problem that influence the user-interface requirements
  - missing, required or unnecessary facts or information
  - success criteria to evaluate the personal, social and economic impacts of the solution.
- Evaluate information and ideas.
- Communicate using
  - digital technologies–specific language
  - language conventions, textual features, such as annotations, paragraphs and sentences, and referencing conventions to convey information to particular audiences about digital solutions
  - sketches or diagrams to present information and ideas about the problem and programmed digital solutions
  - the modes of visual, written and spoken communication to present data and information about digital solutions.

Topic 2: User experiences and interfaces

- Recognise and describe
  - the meaning and importance of user experience
  - useability principles including accessibility, effectiveness, safety, utility and learnability.
- Explore existing user interfaces to
  - identify pitfalls and useful solutions
  - determine how user characteristics influence the user-interface requirements and user experience for problems and solutions in relation to the useability principles.
- Explore and use the elements and principles of visual communication
  - elements are limited to space, line, colour, shape, texture, tone, form, proportion and scale
  - principles are limited to balance, contrast, proximity, harmony, alignment, repetition and hierarchy.
- Symbolise ideas for a user interface using sketches, diagrams, schematic diagrams or mock-ups.
- Generate user interfaces by investigating and applying useability principles.
- Evaluate and make recommendations about user interfaces based on useability principles

Topic 3: Algorithms and programming techniques

- Recognise and describe programming syntax and rules.
- Understand that simple algorithms consist of input, process and output at various stages.
- Understand that each programming language has a unique set of features and constructs that make it suitable for different types of tasks.
- Understand and use the basic algorithm constructs including
  - assignment: used to store the value of an expression into a variable
  - sequence: a number of instructions processed one after the other
  - selection: the next instruction to be executed depends on a 'condition'
  - condition: a logical expression that evaluates to true or false
  - iteration: a number of instructions are repeated
  - modularisation: used for reducing the complexity of a system by deconstructing into more or less independent units or modules.
- Develop algorithms using pseudocode by
  - identifying and describing the steps and their behaviour in the algorithm
  - identifying and explaining the algorithmic steps required for a programmed solution.
- Symbolise interrelationships with sketches and diagrams.

- Recognise, describe and use the five basic features of programming:
  - variables
  - control structures
  - data structures
  - syntax
  - libraries and classes.
- Recognise, describe and use good programming practices, including
  - dependability
  - efficiency
  - testing and debugging
  - error correction
  - coding conventions including
    - commenting
    - consistent naming conventions
    - code simplicity
    - portability.
- Identify and describe
  - the purpose of code syntax and rules
  - the scope and use of local and global variables
  - code object/event triggers and their effect on user interfaces.
- Explore
  - programming development tools to understand how to use them effectively
  - the use of a procedural text-based language for
    - writing and modifying code and using existing code blocks or statements
    - interpreting programming language rules and syntax
    - analysing and critiquing the end result of code statements using input or output evidence, i.e. runtime evidence
  - functions and procedures with efficient and maintainable code that
    - includes reuseable coded components
    - responds to keyboard and mouse events
    - uses variables, selection structures, counted loops, while loops and single, multi-branch and nested conditional logic/statements
    - uses operators, including arithmetic (+, –, *, /, integer, modulus, exponent), comparison (<, >, <=, >=, equal, not equal) and logical (AND, OR, NOT)
  - the purpose of code statements by writing code and using existing code blocks or statements
  - object/event triggers and explain their effect/s on user interfaces.
- Communicate and clarify knowledge and understanding about the purpose of code statements using code comments.

Topic 4: Programmed solutions

- Use a procedural text-based language to
  - apply the use of operators, including
    - arithmetic: +, –, *, /, integer, modulus, exponent
    - comparison: <, >, <=, >=, equal, not equal
    - logical: AND, OR, NOT
  - output information to the screen in text-based or visual formats
  - generate
    - components of a solution by using existing code or writing new code statements
    - modified code in response to new or existing information
    - functions/procedures with efficient and maintainable code that includes reuseable code blocks or statements and responses to keyboard and mouse events
    - selection structures, counted loops, while loops, and single, multi-branch and nested conditional logic statements
    - local and global variables
    - a prototype digital solution in response to a problem
  - test inputs, outputs and processes
  - evaluate and make recommendations about
    - the use of programming language rules and syntax for a given problem
    - algorithmic steps using debugging / testing processes, e.g. desk checks
    - the effectiveness of algorithms
    - the end result of code statements using input or output evidence
    - the user interface based on useability principles including accessibility, effectiveness, safety, utility and learnability
    - the solution and its components by testing to identify errors using computational thinking processes, e.g. debugging techniques
    - the personal, social and economic impacts of the solution
    - the implemented solution against success criteria, maintainability and useability principles.

Unit 2: Application and data solutions

In Unit 2, students are required to engage with and learn subject matter using the various phases of the problem-solving process. Students will optimise a given database and use programming skills acquired in Unit 1 to generate a solution that interacts with an existing database via structured query language (SQL). Students will plan, develop and generate the interface and code to enable the user to insert, update, retrieve and delete data using an existing database via SQL. Prior to inserting the data, the system will validate the data being entered to ensure its integrity and reliability for use and storage. Retrieved data will be displayed to the user in an appropriate format, such as text or a symbolic visual form.

Students are required to understand the structure of a database, along with how primary and foreign keys and data types affect the performance of the database. Students will evaluate the security, privacy and ethical effects of storing data in databases from individual, organisational and government perspectives.

Unit objectives

1. Recognise and describe programming features, data and useability principles, and data management processes.
2. Symbolise and explain information, ideas and data flow relationships within and between systems related to digital problems.
3. Analyse problems and information related to the selected technology context.
4. Determine solution requirements and success criteria of a digital problem.
5. Synthesise information and ideas to determine possible digital solutions.
6. Generate user interface and programmed components of the prototype digital solution.
7. Evaluate impacts, components and solutions against criteria to make refinements and justified recommendations.
8. Make decisions about and use mode-appropriate features, language and conventions for particular purposes and contexts.

Subject matter

Topic 1: Data-driven problems and solution requirements

- Understand the nature of data-driven problems.
- Analyse problems associated with data insertion, including variations in data formats, data structures, validation rules and data requirements.
- Determine manageable aspects of a problem through decomposition, pattern recognition and analysis of
  - user requirements
  - programming options including
    - interactivity, e.g. user input and presentation of data
    - data models, and storage and output requirements.
- Recognise data types, constraints, and primary and foreign keys.
- Recognise and describe useability principles including accessibility, effectiveness, safety, utility and learnability.
- Symbolise the links between external entities, data sources, data flow, processes and data storage in annotated context diagrams or data flow diagrams.
- Develop algorithms using pseudocode.
- Explore and communicate the personal, social and economic impacts of storing data in a database for individuals, organisations and governments.
- Explore potential impacts of emerging technologies on data storage, e.g. machine learning, deep learning, neural networks, natural language processing.
- Explore and use the elements and principles of visual communication
  - elements are limited to space, line, colour, shape, texture, tone, form, proportion and scale
  - principles are limited to balance, contrast, proximity, harmony, alignment, repetition and hierarchy.
- Communicate using
  - digital technologies–specific language
  - language conventions, textual features such as annotations, paragraphs and sentences, and referencing conventions to convey information to particular audiences about digital solutions
  - sketches or diagrams to present information and ideas about the problem and programmed digital solutions
  - the modes of visual, written and spoken communication to present data and information about digital solutions.

Topic 2: Data and programming techniques

- Recognise the elements needed for a data-driven solution, including
  - scope of given problems
  - constraints and limitations
  - programming requirements, e.g. SQL and algorithms
  - system requirements, e.g. platforms, connections, hardware and data stores
  - the data that is required from real-world data sources, e.g. files, peripheral devices, online sources and users
  - the personal, social and economic impacts of storing data in databases for individuals, organisations and governments.
- Understand
  - the difference between data, information and wisdom
  - that data-driven programming is typically applied to streams of structured data for filtering, transforming, aggregating (such as computing statistics), or calling other programs
  - SQL syntax and use SQL statements to solve a problem
  - that simple algorithms consist of input, process and output at various stages
  - that data is organised in tabular form and the skills and knowledge used to normalise and link tables together
  - the reasons and methods of database structure modification to third normal form (3NF).
- Interpret the structure of a database represented by a relational schema (RS) to determine the relationship between data.
- Explain data principles including
  - acquisition
  - organisation, i.e. using appropriate naming conventions, data formats and structures
  - representation
  - integrity
  - anomalies
  - redundancy
  - security.
- Explain
  - the difference between data validation and data verification
  - referential integrity, normalisation and third normal form, relational database management system
  - the difference between primary key and foreign key
  - relations (tables) including rows; columns; primary, secondary and foreign keys; nulls; and views within a database management system.

- Symbolise
  - ideas for user interface and interconnecting systems using sketches, diagrams or mock-ups
  - data flow through a system using data flow diagrams.
- Analyse and structure data and data stores to reduce redundancy and ensure completeness, consistency and integrity for use and storage.
- Apply data management processes, e.g. encryption, consistency, searching, pattern recognition and de-identification.
- Understand and use the basic constructs of an algorithm including assignment, sequence, selection, condition, iteration and modularisation.
- Develop well-ordered and unambiguous algorithms using pseudocode for
  - a program that processes data for insertion into a database or manipulates or displays retrieved data
  - user interaction, data validation and data presentation.
- Communicate and clarify knowledge and understanding about the purpose of code statements using code comments.

Topic 3: Prototype data solutions

- Identify the success criteria to plan the user interface and programmed components of proposed solutions.
- Determine appropriate data types, constraints, and primary and foreign keys.
- Evaluate and modify a database structure to third normal form (3NF).
- Use SQL statements including
  - SELECT, WHERE, GROUP BY, HAVING, ORDER BY, COUNT, MIN, MAX, AVG, IN, inner-joins and sub-queries to retrieve appropriate data from existing databases
  - CREATE, INSERT, UPDATE and DELETE to create database tables and views, and modify stored data.
- Generate a prototype digital solution to access, manipulate and display data that
  - enables data to be inserted, updated, retrieved and deleted from single and multiple tables
  - validates the data to be entered for reliability to ensure that the data is valid for use and storage
  - includes user interfaces that will enable the insertion, updating and selection of data from/to an SQL database.
- Test the SQL and programmed components of the prototype digital solution for reliability, maintainability and efficiency.
- Test the user interfaces against useability principles.
- Evaluate
  - algorithmic steps using debugging / testing processes, e.g. desk checks
  - data quality using the success criteria of accuracy and completeness
  - the prototype digital solution against success criteria.

Unit 3: Digital innovation

In Unit 3, students are required to engage with and learn subject matter using the various phases of the problem-solving process. Students analyse end-user needs, and use the knowledge and skills of problem-solving, computational, design and systems thinking. They determine data, programming and user experience requirements, using available resources to generate components and prototyped digital solutions. Students do this through one of the technology contexts: web applications, mobile applications, interactive media, or intelligent systems.

Unit objectives

1. Recognise and describe programming features, digital system, interface components, and useability principles.
2. Symbolise and explain programming information, ideas and interrelationships between data structures and user experiences.
3. Analyse problems and information related to the selected technology context.
4. Determine solution requirements, and success criteria of a digital problem.
5. Synthesise information, ideas and data elements to determine possible solutions for user interface and programmed components.
6. Generate user interfaces and programmed components of the prototype digital solution.
7. Evaluate impacts, components and a solution against criteria to make refinements and justified recommendations.
8. Make decisions about and use mode-appropriate features, language and conventions for particular purposes and contexts.

Subject matter

Topic 1: Interactions between users, data and digital systems

- Explore and analyse the meaning and importance of innovation and the personal, business and social opportunities presented by innovation.
- Explore the impact of emerging technologies on innovation, e.g. machine learning, deep learning, neural networks, natural language processing.
- Recognise and describe components of a digital solution appropriate to the technology context selected.

Technology context	Web applications	Mobile applications	Interactive media	Intelligent systems
• components on which to focus	• server-side components including web server, DBServer and pre-processing components such as PHP • client-side components including web browser and user device • data components such as database structure • internal data structures such as arrays, lists and dictionaries	• user-interface components such as user hardware and functionality to provide input and output • program components such as objects, event handlers • data resources such as external libraries and internal application data structures	• user-interface components such as user hardware and peripheral devices used for input and output • program components such as objects, event handlers and multimedia assets • external data stores such as file structures or object libraries • internal data structures such as arrays, lists and dictionaries	• sensors • actuators • user-interface components • analogue and digital input/output data streams • administrative interface components • network hardware and protocols • internal data structures appropriate to the hardware storage code library selected

- Analyse a problem to identify and explain the
  - components of a system, e.g. hardware, software, data, network, processes, users, interfaces, protocols and standards
  - observable interactions, e.g. human-human interactions through communication channels such as social media platforms, email, instant messaging, video conferencing
  - inputs and outputs
  - control mechanism
  - processes and interactions using logical diagrams and consistent symbols.
- Symbolise and explain
  - useability principles, including accessibility, effectiveness, safety, utility and learnability
  - a variety of interfaces
  - data flow through a system using data flow diagrams.
- Symbolise, explain and use advanced data processes, including table joins, referential integrity, redundancy reduction and anomaly updating.

- Explore
  - flexible development methods to support a variety of user profiles
  - methods of synthesising user interface, processing and data components to generate a prototype digital solution.
- Explore and use the elements and principles of visual communication
  - elements are limited to space, line, colour, shape, texture, tone, form, proportion and scale
  - principles are limited to balance, contrast, proximity, harmony, alignment, repetition and hierarchy.
- Determine possible personal, social and economic impacts.
- Appraise user interfaces against useability principles.

Topic 2: Real-world problems and solution requirements

- Explore programming development tools to understand how to use them effectively.
- Explore project management tools and processes to understand how to use them effectively, e.g. tasks, milestones, dependencies, stakeholders, resources.
- Analyse problems and information to determine
  - manageable aspects of the problem
  - a specific aspect of the problem to develop
  - scope of the problem
  - constraints and limitations of the environment
  - requirements of the solution
  - success criteria.
- Describe interactions in terms of inputs, processes and outputs, e.g. machine–human interactions such as smartphone applications and chatbots, human-machine interactions such as touchscreens, voice commands and gestures.
- Explore data sources to understand relational and flat file data structures.
- Evaluate ideas using innovation and collaboration.
- Recognise and compare different file formats and data structures appropriate to the context.
- Determine file formats and data structures appropriate to the technology context.
- Analyse modularity and readability of program modules.
- Recognise and use
  - the basic constructs of an algorithm including assignment, sequence, selection, condition, iteration and modularisation
  - pseudocode to communicate algorithmic steps.
- Understand that simple algorithms consist of input, process and output.

- Describe well-ordered and unambiguous algorithms using pseudocode for
  - procedural code that processes data for insertion into a database or manipulates or displays retrieved data
  - user interaction, data validation and data presentation.
- Explain code steps using comment syntax appropriate to the programming language.
- Apply
  - computational thinking processes, e.g. creating, debugging, persevering and collaborating to identify possible algorithmic approaches
  - data algorithms for cleaning and merging data sources and iterating through data records.
- Use generic pseudocode suitable for a variety of programming languages to communicate requirements for programmed components.
- Observe different styles of presenting a technical proposal for a digital solution.
- Communicate a technical proposal for a digital solution through a presentation.
- Communicate using
  - digital technologies–specific language
  - language conventions, textual features such as annotations, paragraphs and sentences, and referencing conventions to convey information to particular audiences about digital solutions
  - sketches or diagrams to present information and ideas about the problem and programmed digital solutions
  - the modes of visual, written and spoken communication to present data and information about digital solutions.

Topic 3: Innovative digital solutions

- Refine ideas for components of a prototype digital solution.
- Demonstrate a prototype of a digital solution.
- Use SQL statements to
  - INSERT, UPDATE and DELETE rows in a database
  - CREATE, DROP and ALTER statements
  - SELECT, WHERE, GROUP BY, HAVING, ORDER BY, sub-selection and inner-joins clauses.
- Develop a conceptual model of a possible solution by applying systems thinking that identifies
  - system boundaries
  - properties
  - inputs and outputs
  - user interface
  - system controls.

- Generate
  - low-fidelity user-interface prototypes appropriate to the digital context by using the elements and principles of visual communication such as sketches, mood boards, storyboards, sitemaps, wireframes and mock-ups
  - simple programs by using programming development tools
  - code that creates, reads, writes, opens and closes a file.
- Generate program modules that
  - interact with users
  - interact with 2D data sources
  - validate data inputs
  - control the interactions in a digital solution.
- Communicate and clarify knowledge and understanding about the purpose of code statements using code comments.
- Synthesise user interface, processing and data components to generate a prototype digital solution.
- Justify selection of relevant data from existing data sources.
- Evaluate
  - the reliability, maintainability, efficiency, effectiveness and useability of algorithms to draw conclusions and make recommendations
  - algorithmic steps using debugging / testing processes, e.g. desk checks
  - user interfaces from existing solutions using heuristic reviews for the useability principles
  - by testing program modules for reliability, maintainability and efficiency using computational thinking processes such as debugging to refine a prototype digital solution.
- Evaluate and refine user interfaces by
  - testing the useability principles, including accessibility, effectiveness, safety, utility and learnability
  - observing and recording user interactions from user experience testing.
- Evaluate against success criteria the
  - user interface and programmed solutions
  - prototype digital solution.

Unit 4: Digital impacts

In Unit 4, students learn how data is shared in both local and global contexts, particularly how digital solutions are increasingly required to exchange data securely and efficiently. Students will understand elements of cybersecurity by exploring the conditions, environment and methods for enabling data to flow between different digital systems. They will analyse data privacy and data integrity risks associated with transferring data between applications and evaluate the personal, social and economic impacts associated with the use and availability of both public and private data. Students will generate an application that simulates the exchange of data between two applications.

Unit objectives

1. Recognise and describe programming features, components of data exchange systems, privacy principles and data exchange processes.
2. Symbolise and explain data structures and specifications, methods for exchanging data, and data-flow relationships within and between systems.
3. Analyse problems and information related to digital systems.
4. Determine solution requirements and success criteria.
5. Synthesise information and ideas to determine possible components of digital solutions.
6. Generate components of the digital solution.
7. Evaluate impacts, components and solutions against success criteria to make refinements and justified recommendations.
8. Make decisions about and use mode-appropriate features, language and conventions for particular purposes and contexts.

Subject matter

Topic 1: Digital methods for exchanging data

- Recognise and describe
  - encryption and authentication strategies appropriate for securing data transmissions and their differences, e.g. two-factor or multi-factor authentication (2FA/MFA) using verification codes vs. biometrics
  - features of symmetric (Data Encryption Standard — DES, Triple DES, AES — Advanced Encryption Standard, Blowfish and Twofish) and assymetric (RSA) encryption algorithms
  - how data compression, encryption and hashing are used in the storage and transfer of data
  - how simple algorithms consist of input, process and output at various stages
  - basic concepts and definitions of emerging technologies, e.g. machine learning, deep learning, neural networks, natural language processing
  - how particular algorithms process data differently, e.g. machine learning, deep learning, natural language processing and reinforcement learning algorithms
  - how useability principles are used to inform solution development
  - how the elements and principles of visual communication inform user interface development.
- Explain
  - Australian Privacy Principles (2014) and ethics applicable to the use of personally identifiable or sensitive data from a digital systems perspective
  - network performance metrics of latency, jitter, and the guarantee and timeliness of delivery
  - network transmission principles, including, protocol standards e.g. TCP/IP, HTTP, FTP and VPN, packet switching, error detection and correction, routing and forwarding, flow and congestion control, quality of service (QoS), security e.g. confidentiality, integrity and availability of data
  - methods for data exchange used to transfer data across networked systems, including REST, JSON and XML, with the assistance of APIs that facilitate data exchange between different systems and applications.
- Analyse and evaluate Caesar, Polyalphabetic (e.g. Vigenere and Gronsfeld), and one-time pad encryption algorithms.
- Describe data using appropriate naming conventions, data formats and structures.
- Symbolise and explain
  - how application sub-systems, e.g. front end, back end, work together to constitute a solution
  - secure data transmission techniques and processes, including the use of encryption, decryption, authentication, hashing and checksums.
- Symbolise
  - representations of a digital solution
  - data flow through a system using data flow diagrams.
- Develop algorithms with pseudocode using the basic constructs including assignment, sequence, selection, condition, iteration and modularisation.

Topic 2: Complex digital data exchange problems and solution requirements

- Identify, describe and determine the scope and usage of local and global variables.
- Analyse problems and information to determine
  - scope of given problems
  - constraints and limitations
  - requirements of the solution components
  - necessary coded modularity and features
  - factors and risks that affect data security, including confidentiality, integrity and availability, and privacy
  - existing code within inbuilt libraries
  - success criteria to appraise the implementation, e.g. protection, security and interactions
  - the potential role of emerging technologies in data exchange solutions, e.g. machine learning.
- Analyse, evaluate and make refinements to data to ensure completeness, consistency and integrity.
- Analyse and explain a system's data process by developing data flow diagrams that link external entities, data sources, processes and data storage.
- Determine manageable aspects of a problem through a decomposition and analysis of
  - constraints
  - risks
  - available tools, code libraries and frameworks
  - data storage and output requirements
  - data interface.
- Determine data sources required to generate data components.
- Develop algorithmic steps with pseudocode.
- Explain the purpose of code and/or algorithm statements using code comments and annotations.
- Communicate using
  - digital technologies–specific language
  - language conventions; textual features such as annotations, paragraphs and sentences; and referencing conventions to convey information to particular audiences about digital solutions
  - sketches or diagrams to present information and ideas about the problem and programmed digital solutions
  - the modes of visual, written and spoken communication to present data and information about digital solutions.

Topic 3: Prototype digital data exchanges

- Synthesise information and ideas to determine prototype components of data exchange solutions.
- Understand and use the basic algorithm constructs including
  - assignment: used to store the value of an expression into a variable
  - sequence: a number of instructions processed one after the other
  - selection: the next instruction to be executed depends on a ‘condition’
  - condition: a logical expression that evaluates to true or false
  - iteration: a number of instructions are repeated
  - modularisation: used for reducing the complexity of a system by deconstructing into more or less independent units or modules.
- Understand and use the five basic features of programming
  - variables
  - control structures
  - data structures
  - syntax
  - libraries and classes.
- Recognise, describe and use good programming practices, including dependability, efficiency, testing, debugging, error correction, coding conventions including commenting, consistent naming conventions, code simplicity and portability.
- Use pseudocode to develop
  - simple Caesar, Polyalphabetic (e.g. Vignere and Gronsfeld), and one-time pad encryption algorithms
  - a well-ordered and unambiguous algorithm to solve defined problems.
- Use a suitable programming environment to
  - implement algorithms using modularisation
  - receive data from an external source, and process and display the data in an appropriate format
  - incorporate existing code libraries (where applicable).
- Generate a prototype digital solution that uses appropriate data formats, e.g. JSON or XML, to exchange data.
- Manipulate data from an external source.
- Use SQL statements including
  - CREATE, DROP and ALTER
  - INSERT and UPDATE
  - SELECT, WHERE, GROUP BY, HAVING, ORDER BY, sub-selection and inner-joins clauses.

- Use programmed methods including
  - sequence
  - selection, i.e. use of single and nested, simple or compound conditions
  - iterations, including nesting or simple or compound conditions
  - code-specific arithmetic comparison and logical operators, including real division, integer division, modulus
  - data types, error-checking functions and conversions
  - structures, including one-dimensional collections, e.g. arrays and lists.
- Evaluate algorithmic steps using desk checks to predict the output for a given input, identify and fix errors (debug) and validate algorithms.
- Evaluate
  - the personal, social and economic impacts of the exchange, storage, accuracy and ownership of data
  - the personal, social and economic impacts of emerging technologies, e.g. artificial intelligence
  - solutions against success criteria
  - solutions by testing to refine their accuracy, reliability, maintainability, efficiency, effectiveness and useability
  - and make justified recommendations related to the security impacts of digital solutions, taking into consideration changes in interactivity and ways information and data are created, used and shared.

Assessment

Internal assessment 1: Technical proposal (25%)

Students generate non-coded low-fidelity prototypes that use an external data source in response to a real-world problem in the selected Unit 3 technology context. They communicate the technical feasibility of the solution through a multimodal presentation.

Assessment objectives

1. Recognise and describe data sources, user interface components and existing solutions.
2. Symbolise user interfaces and explain ideas and interrelationships between proposed data structures and user experiences.
3. Analyse the problem and information related to the selected technology context.
4. Determine data, programming and user experience requirements of the identified problem and success criteria.
5. Synthesise information and ideas to determine the possible solutions for data, user interface and algorithmic components.
6. Generate a low-fidelity non-coded prototype digital solution.
8. Make decisions about and use mode-appropriate features, language and conventions for written and spoken communication for a technical audience.

Specifications

This task requires students to:

- recognise and describe
  - data sources
  - user-interface components
  - existing solutions to similar problems
- symbolise ideas for user interfaces using one or more of constructed sketches, annotated diagrams, images or screenshots
- explain
  - ideas
  - interrelationships between proposed data structures and user experiences
- analyse the problem to identify
  - scope of the problem
  - constraints and limitations
  - possible personal, social and economic impacts
  - possible solutions
  - data, programming and user-interface relationships

- analyse information to determine
  - user experience requirements from the user perspective
  - programming requirements from the developer perspective
  - required data
  - success criteria
- synthesise information and ideas to determine the possible solutions for
  - data of the proposed solution
  - user interface/s
  - algorithmic components
- generate a low-fidelity (non-coded) prototype solution including
  - data
  - user interface
- communicate
  - information and ideas to inform a technical audience
  - the technical feasibility of generating the prototype solution, including the technical aspects of the development process, e.g. algorithms, selection and justification of development tools, user-interface sketches, user-experience requirements.

It is recommended that this task is designed so that students can develop a response in approximately 15 hours of class time.

Conditions

- Students can develop their responses in class time and their own time.
- This is an individual task.

Response requirements

Visual, spoken and/or written (including low-fidelity non-coded prototype): up to 10 minutes, including annotations of up to 2000 words

Mark allocation

Criterion	Assessment objectives	Marks
Comprehending	1, 2	5
Analysing	3, 4	6
Synthesising	5	6
Generating	6	6
Communicating	8	2
	Total marks:	25

Instrument-specific marking guide

Comprehending	Marks
The student response has the following characteristics:	
• discerning recognition and description of – data sources – user-interface components – existing solutions • adept symbolisation of – user interfaces • discerning explanation of – ideas – interrelationships between proposed data structures and user experiences	4–5
• adequate recognition and description of – data sources – user-interface components – existing solutions • competent symbolisation of – user interfaces • adequate explanation of – ideas – interrelationships between proposed data structures and user experiences	2–3
• makes statements about features of – data – programming – user interface • incomplete symbolisation of – user-interfaces • superficial explanation of – ideas – interrelationships.	1
The student response does not match any of the descriptors above.	0

Analysing	Marks
The student response has the following characteristics:	
• insightful analysis of the problem and contextual information to identify features and relationships of − data − programming − user-interface • astute determination of − programming requirements − user-experience requirements − success criteria	5–6
• adequate analysis of the problem and contextual information to identify features and relationships of − data − programming − user interface • reasonable determination of − programming requirements − user-experience requirements − success criteria	3–4
• superficial analysis of the problem or information to identify some features or relationships of − data − programming − user-interface • vague determination of − programming or user-experience requirements − success criteria.	1–2
The student response does not match any of the descriptors above.	0

Synthesising	Marks
The student response has the following characteristics:	
• logical synthesis of information and ideas to determine the possible solutions for − user interfaces − algorithms − data	5–6
• adequate synthesis of information and ideas to determine the possible solutions for − user interfaces − algorithms − data	3–4
• simple synthesis of information or ideas to determine the possible solutions for − user interfaces − algorithms − data.	1–2
The student response does not match any of the descriptors above.	0

Generating	Marks
The student response has the following characteristics:	
• effective generation of a non-coded low-fidelity prototype digital solution for – user interfaces – data	5–6
• adequate generation of a non-coded low-fidelity prototype digital solution for – user interfaces – data	3–4
• generation of elements of the non-coded low-fidelity prototype digital solution for some – user interfaces – data.	1–2
The student response does not match any of the descriptors above.	0

Communicating	Marks
The student response has the following characteristics:	
• effective decision-making about, and fluent use of – written, visual and/or spoken features to communicate about a solution – language for a technical audience – grammatically accurate language structures – referencing conventions	2
• inadequate decision-making about, and inconsistent use of – written, visual and/or spoken features – suitable language – grammar and language structures – referencing conventions.	1
The student response does not match any of the descriptors above.	0

Internal assessment 2: Digital solution (25%)

Students generate an innovative digital solution to a real-world problem in the selected Unit 3 technology context. Students document the application of the problem-solving process in response to a stimulus document supplied by the teacher.

Assessment objectives

5. Synthesise information and ideas to determine the possible solutions for data, user interface and programmed components for a digital solution.
6. Generate user interfaces and programmed components of the digital solution.
7. Evaluate components and the digital solution against success criteria to make refinements and justified recommendations and evaluate impacts.
8. Make decisions about and use mode-appropriate features, written language and conventions for a technical audience.

Specifications

This task requires students to:

- synthesise ideas and information to determine the possible solutions for
  - user interfaces
  - data and data repositories
  - programmed components
- generate
  - programmed components for the prototype digital solution demonstrating selection, iteration, user input, data output
  - a prototype digital solution by combining the user interface, data and programmed components
- evaluate a prototype of the digital solution against criteria, including
  - the accuracy and maintainability of programmed components
  - user experience
- make refinements and recommendations for current and future improvements, justified by user feedback and testing
- evaluate the personal, social and economic impacts of the generated digital solution within the context of the real-world problem.

It is recommended that this task is designed so that students can develop a response in approximately 15 hours of class time.

Stimulus specifications

The teacher provides a stimulus that includes:

- functional requirements (define what the system is supposed to do or accomplish)
  - interactions between machines and humans
- non-functional requirements (define how well a system performs its functions)
  - user experience
    - useability
    - visual communication
- end-user profile/s and/or proto-personas
- data sources and information related to data repositories.

Conditions

- Students can develop their responses in class time and their own time.
- This is an individual task.

Response requirements

- Visual and written (including mind maps, models, sketches, diagrams, tables, images, screenshots, schemas): up to 10 A4 pages, including annotations of up to 1500 words
- Visual and spoken video: up to 2 minutes, demonstrating the functionality of
  - the user interface
  - data and programmed components

Mark allocation

Criterion	Assessment objectives	Marks
Synthesising	5	7
Generating	6	9
Evaluating	7	7
Communicating	8	2
	Total marks:	25

Instrument-specific marking guide

Synthesising	Marks
The student response has the following characteristics:	
• logical synthesis of relevant information and ideas to determine the possible solutions for – user interfaces – data and data repositories – programmed components	6–7
• adequate synthesis of information and ideas to determine the possible solutions for – user interfaces – data and data repositories – programmed components	4–5
• simple synthesis of information or ideas to determine the possible solutions for – user interfaces – data and data repositories – programmed components	2–3
• unclear combination of information or ideas to determine data, data repositories, user interface or programmed components.	1
The student response does not match any of the descriptors above.	0

Generating	Marks
The student response has the following characteristics:	
• sophisticated generation of a prototype digital solution combining – user interface components – data components – programmed components	8–9
• effective generation of a prototype digital solution combining – user interface components – data components – programmed components	6–7
• adequate generation of a prototype digital solution combining – user interface components – data components – programmed components	4–5
• basic generation of a prototype digital solution combining – user interface components – data components – programmed components	2–3
• generation of elements of the prototype digital solution.	1
The student response does not match any of the descriptors above.	0

Evaluating	Marks
The student response has the following characteristics:	
• critical evaluation of features and components against criteria including – user experience – programmed components • critical evaluation of impacts • effective refinements and recommendations justified by – user feedback – testing	6–7
• feasible evaluation of features and components against criteria including – user experience – programmed components • feasible evaluation of impacts • adequate refinements and recommendations justified by – user feedback – testing	4–5
• superficial evaluation of – user experience – programmed components – impacts	2–3
• identification of a change to an idea or a solution.	1
The student response does not match any of the descriptors above.	0

Communicating	Marks
The student response has the following characteristics:	
• effective decision-making about, and fluent use of – written, visual and/or spoken features to communicate about a solution – language for a technical audience – grammatically accurate language structures – referencing conventions	2
• inadequate decision-making about, and inconsistent use of – written, visual and/or spoken features – suitable language – grammar and language structures – referencing conventions.	1
The student response does not match any of the descriptors above.	0

Internal assessment 3: Digital solution (25%)

Students generate an innovative digital solution to a real-world problem with a focus on data security and impacts in any of the four technology contexts: web application, mobile application, interactive media or intelligent systems. Students document the application of the problem-solving process in response to a stimulus supplied by the teacher.

Assessment objectives

5. Synthesise information and ideas to determine the possible solutions for data, user interface and programmed components for a secure digital solution.
6. Generate user interfaces and programmed components of the digital solution.
7. Evaluate components and the digital solution against success criteria to make refinements and justified recommendations and evaluate impacts.
8. Make decisions about and use mode-appropriate features, written language and conventions for a technical audience.

Specifications

This task requires students to:

- synthesise ideas and information to determine the possible solutions for secure
  - user interfaces
  - data and data repositories
  - programmed components
- generate
  - programmed components for the prototype digital solution demonstrating selection, iteration, user input, data output
  - a prototype digital solution by combining the user interface, data and programmed components
- evaluate a prototype of the digital solution against success criteria, including
  - the accuracy and maintainability of programmed components
  - user experience
- make refinements and recommendations for current and future improvements, justified by user feedback and testing
- evaluate the personal, social and economic impacts on data security and privacy, within the context of the real-world problem and generated digital solution.

It is recommended that this task is designed so that students can develop a response in approximately 15 hours of class time.

Stimulus specifications

The teacher provides a stimulus document that includes:

- functional requirements (define what the system is supposed to do or accomplish)
  - interactions between machines and humans
- non-functional requirements (define how well a system performs its functions)
  - user experience
    - useability
    - visual communication
- end-user profiles and/or proto-personas
- data sources and information related to data repositories.

Conditions

- Students can develop their responses in class time and their own time.
- This is an individual task.

Response requirements

- Visual and written (including mind maps, models, sketches, diagrams, tables, images, screenshots, schemas): up to 10 A4 pages, including annotations of up to 1500 words
- Visual and spoken video: up to 2 minutes, demonstrating the functionality of
  - the user interface
  - data and programmed components

Mark allocation

Criterion	Assessment objectives	Marks
Synthesising	5	7
Generating	6	9
Evaluating	7	7
Communicating	8	2
	Total marks:	25

Instrument-specific marking guide

Synthesising	Marks
The student response has the following characteristics:	
• logical synthesis of relevant information and ideas to determine the possible solutions for secure – user interfaces – data and data repositories – programmed components	6–7
• adequate synthesis of information and ideas to determine the possible solutions for secure – user interfaces – data and data repositories – programmed components	4–5
• simple synthesis of information or ideas to determine the possible solutions for secure – user interfaces – data and data repositories – programmed components	2–3
• unclear combination of information or ideas to determine secure data, data repositories, user interface or programmed components.	1
The student response does not match any of the descriptors above.	0

Generating	Marks
The student response has the following characteristics:	
• sophisticated generation of a prototype digital solution combining – user interface components – data components – programmed components	8–9
• effective generation of a prototype digital solution combining – user interface components – data components – programmed components	6–7
• adequate generation of a prototype digital solution combining – user interface components – data components – programmed components	4–5
• basic generation of a prototype digital solution combining – user interface components – data components – programmed components	2–3
• generation of elements of the prototype digital solution.	1
The student response does not match any of the descriptors above.	0

Evaluating	Marks
The student response has the following characteristics:	
• critical evaluation of features and components against criteria including – user experience – programmed components • critical evaluation of impacts • effective refinements and recommendations justified by – user feedback – testing	6–7
• feasible evaluation of features and components against criteria including – user experience – programmed components • feasible evaluation of impacts • adequate refinements and recommendations justified by – user feedback – testing	4–5
• superficial evaluation of – user experience – programmed components – impacts	2–3
• identification of a change to an idea or a solution.	1
The student response does not match any of the descriptors above.	0

Communicating	Marks
The student response has the following characteristics:	
• effective decision-making about, and fluent use of – written, visual and/or spoken features to communicate about a solution – language for a technical audience – grammatically accurate language structures – referencing conventions	2
• inadequate decision-making about, and inconsistent use of – written, visual and/or spoken features – suitable language – grammar and language structures – referencing conventions.	1
The student response does not match any of the descriptors above.	0

External assessment: Examination — combination response (25%)

External assessment is developed and marked by the QCAA. The external assessment in Digital Solutions is common to all schools and administered under the same conditions, at the same time, on the same day.

Assessment objectives

1. Recognise and describe user-experience elements, programming features, components of data exchange systems, privacy principles and data exchange processes.
2. Symbolise and explain programming ideas, data specifications, data exchange processes, and data flow within and between systems.
3. Analyse problems and information related to a digital problem.
5. Synthesise information and ideas to determine possible low-fidelity components of secure data exchange solutions.
7. Evaluate components and solutions against criteria to make refinements and justified recommendations and evaluate impacts.

Specifications

This examination:

- consists of a number of different types of questions relating to Unit 4
- may ask students to respond using
  - multiple choice
  - short and extended response in sentences or paragraphs
- may ask students to
  - sketch, draw and/or diagrams
  - write and calculate using algorithms
  - interpret unseen stimulus materials.

Conditions

- Time allowed
  - Perusal time: 5 minutes
  - Working time: 120 minutes
- Students may use a QCAA-approved non-programmable scientific calculator.

Glossary

The syllabus glossary is available at www.qcaa.qld.edu.au/downloads/senior-qce/common/snr_glossary_cognitive_verbs.pdf.

References

American National Standards Institute symbols

Australian Curriculum 2017, *Structure*, www.australiancurriculum.edu.au/f-10-curriculum/technologies/digital-technologies.

Australian Curriculum 2017b, *Glossary*, https://www.australiancurriculum.edu.au/f-10-curriculum/technologies/glossary.

Australian Government Office of the Australian Information Commissioner 2013, *Privacy Fact Sheet 17: Australian Privacy Principles*, www.oaic.gov.au/individuals/privacy-fact-sheets/general/privacy-fact-sheet-17-australian-privacy-principles.

Ferguson, D 2009, *Development of Technology Education in New Zealand Schools 1985–2008*, https://www.technology.tki.org.nz/content/download/244/1153/file/DevelopmentofTechEducation-Sept09.

Gane, C & Sarson, T 1979, *Structured Systems Analysis*, Prentice Hall, New Jersey.

Marzano, RJ & Kendall, JS 2008, *Designing and Assessing Educational Objectives: Applying the new taxonomy*, Corwin Press, USA.

Marzano, R J & Kendall, J S 2007, *The New Taxonomy of Educational Objective*s (2nd edn), Corwin Press, USA.

NSW Board of Studies 2017, *A Guide to Coding and Computational Thinking across the Curriculum*, www.k6.boardofstudies.nsw.edu.au/wps/portal/nesa/k-10/learning-areas/technology/coding-across-the-curriculum.

Oxford University Press 2017, *Oxford Dictionaries*, www.oed.com.

Pastel, R 2017 *CS4760 & CS5760: Human-Computer Interactions & Usability*, http://cs4760.csl.mtu.edu/2017/lectures/usability.

Preece, J, Rogers, Y & Sharp, H 2002, *Interaction Design: Beyond human-computer interaction*, John Wiley & Sons, New York.

Victorian Curriculum and Assessment Authority 2014, *Computing: Study design: Accreditation period 2016–2019*, www.vcaa.vic.edu.au/Documents/vce/computing/ComputingSD-2016.pdf.

Wenzel, K 2017, *Database Normalization Explained in Simple English*, www.essentialsql.com/get-ready-to-learn-sql-database-normalization-explained-in-simple-english.

Version history

Version	Date of change	Information
1.0	January 2024	Released for familiarisation and planning (with implementation starting in 2025)