Government of Western Australia
School Curriculum and Standards Authority

COMPUTER SCIENCE

ATAR COURSE

Year 11 syllabus

Acknowledgement of Country

Kaya. The School Curriculum and Standards Authority (the Authority) acknowledges that our offices are on Whadjuk Noongar boodjar and that we deliver our services on the country of many traditional custodians and language groups throughout Western Australia. The Authority acknowledges the traditional custodians throughout Western Australia and their continuing connection to land, waters and community. We offer our respect to Elders past and present.

Important information

This syllabus is effective from 1 January 2024.

Users of this syllabus are responsible for checking its currency.

Syllabuses are formally reviewed by the School Curriculum and Standards Authority (the Authority) on a cyclical basis, typically every five years.

Disclaimer

Any resources such as texts, websites and so on that may be referred to in this document are provided as examples of resources that teachers can use to support their learning programs. Their inclusion does not imply that they are mandatory or that they are the only resources relevant to the course.

Content

Rationale

The Computer Science ATAR course builds on the core principles, concepts and skills developed in the Digital Technologies subject in previous years. Students utilise and enhance established analysis and algorithm design skills to create innovative digital solutions to real-world problems. In the process, students develop computational, algorithmic and systems thinking skills which can be successfully applied to problems across domains outside Information Technology. In addition to the development of software, the essential concepts of networking, data management and cyber security are explored. With the vast amounts of data collected in our increasingly digital world, especially in the information-intensive business and scientific disciplines, data management is becoming essential. Similarly, with more and more devices connecting to the internet, cyber security is a major issue for society and the world continues to look for new, young experts to emerge in this field.

Ethical considerations, security requirements and legal factors affect society as a whole and their influence and impact on the development of digital solutions are explored.

This course provides students with options in a range of post-school pathways. The course has been designed to meet the expectations of tertiary institutions and students will be well prepared for further study in university and TAFE courses. It provides a sound understanding of computer science to support students pursuing further studies and employment in other areas, including Science, Technology, Engineering, Mathematics and Business, all of which are underpinned and driven by advances in Computer Science.

Course Aims

The Computer Science ATAR course aims to develop students':

- skills in
  - designing, maintaining, adapting and producing relational databases and digital solutions
  - solving problems through the use of algorithms, data structures and programming languages
  - assessing cybersecurity issues within a digital environment to apply appropriate responses
- understanding of the design, application and interactions of data and software in digital systems through the creation and maintenance of relational databases, network data transmission and programming constructs
- understanding of how to apply a technology process accurately to develop a digital solution
- understanding of the interrelationships between the development and use of digital solutions, for individuals and societies in relation to the legal and ethical implications of software design, data management and cyber threats.

Organisation

This course is organised into a Year 11 syllabus and a Year 12 syllabus. The cognitive complexity of the syllabus content increases from Year 11 to Year 12.

Structure of the syllabus

The Year 11 syllabus is divided into two units, each of one semester duration, which are typically delivered as a pair. The notional time for each unit is 55 class contact hours.

Each unit includes:

- a unit description – a short description of the purpose of the unit
- unit content – the content to be taught and learned.

Technology Process

A major focus of the course is the creation of systems and digital solutions to specific problems. In creating solutions, it is expected that students will use a structured development process to guide their approach. This development process is iterative in nature and involves four phases – investigating the problem, developing ideas and designing a solution, developing a solution and evaluating the effectiveness of the solution.

Investigate: students will define the problem. They will deconstruct the problem and develop a set of criteria that can be used to design and evaluate a solution to the problem.

Design: students will use the criteria they developed in their investigation to design a suitable digital solution to the problem they have defined.

Develop: students will use their programming and database skills to develop a suitable digital solution that will meet the criteria they have established to solve the problem.

Evaluate: students will evaluate the effectiveness of their solution in solving the problem that they defined during their analysis. This will allow students to refine their ideas and make partial or minor changes to their solution to improve the user experience and/or the technical quality of their solution.

Unit 1 – Design and development of programming and network solutions

In this unit, students gain knowledge and skills to create software solutions. They use algorithms and structured programming to design and implement software solutions for a range of problems. They consider the complex interactions between users, developers, the law, ethics and society when computer systems are used and developed. Students learn about network communications and the transfer of data through a network.

Unit 2 – Design and development of database solutions and cyber security considerations

In this unit, students learn about the design concepts and tools used to develop relational database systems. Students gain knowledge and skills to create database solutions and create queries to extract relevant information. Students consider the security of network communications, exploring a range of threats and measures used to keep networks secure. Students examine attitudes and values involved in the creation and

use of computer-based systems and their effect on society. They examine the ethical and legal obligations of the user and developer in the collection and storage of data.

Organisation of content

The unit content includes both theoretical aspects (Knowledge) and practical aspects (Skills).

The course is divided into four content areas.

Unit 1 is divided into two content areas:

- Programming
- Network communications.

Unit 2 is divided into two content areas:

- Cyber security
- Data management.

Programming

Solutions develop solutions to specific problems using both pseudocode and a programming language. They examine the use of various simple and complex data types with the basic constructs of sequence, selection and iteration. Complex problems are analysed and broken down into small, self-contained units for which students create functions, with parameter passing. In this unit, students consider good programming practices and the suitability of an algorithm, including the use of standard algorithms to complete common tasks.

Network communications

Students explore the communication models and protocols underpinning the transfer of data in local networks and the internet. They explore purpose of layers of a network and the components that operate within them, taking into account factors that affect the design of the network. They investigate design and creation of secure and efficient networks.

Cyber security

Students explore the communication models and protocols underpinning the transfer of data in local networks and the internet. They examine methods of keeping network communications secure over an open connection, such as the internet. Students consider the legal requirements and ethical responsibilities of developers, organisations and penetration testers with respect to the management of data and networks.

Data management

Students examine the organisation of data into separate entities using a relational database and the process of normalisation. They explore various methods of representing and organising data and develop a working database solution using SQL. The legal and ethical responsibilities with regards to the collection and storage of data are considered.

Programming languages

Python is the prescribed programming language for the Computer Science ATAR course and will be used in ATAR examination questions related to programming. In school based assessments, schools may choose to also use an object-oriented programming language of their choice for the:

- development of a purpose-designed software solution
- design, creation, modification, testing, evaluation and documentation of programs
- writing, interpretation, testing and debugging of code
- use and development of a user interface
- use of control structures, including sequence, selection and iteration
- construction and use of data structures, including arrays and dictionaries
- design and implementation of data validation techniques
- application of structured programming methods using modularisation.

There is no requirement within this course to create a graphical user interface.

Database management systems

There is no prescribed database management system for the Computer Science ATAR course, but SQLite is recommended. If an alternative to SQLite is chosen, it must meet the assessment requirements for this syllabus that students are required to use a database management system (DBMS) that enables the:

- development of a purpose-designed database solution
- design, creation, modification, testing and evaluation of a database solution
- creation of tables and queries.

Note: the course does not require the development of a graphical user interface for the database system, although some method of allowing the user to interact with the database is required.

The database management systems should provide the student with opportunity to:

- create a working relational database
- construct simple queries using SQL within one or two tables
- construct queries across multiple tables using a database tool.

Progression from the Year 7–10 curriculum

This syllabus continues to develop student learning around the knowledge, understandings and skills within the Year 7–10 Digital Technologies curriculum and focuses on the components of digital systems (software, hardware and networks) and their use; the representation of data; and how data are represented and structured symbolically.

This syllabus also continues to develop the students' skills with the production of digital solutions through: collecting, managing and analysing data, defining problems, designing solutions, implementing and evaluating solutions, and communicating, collaborating and managing projects.

Representation of the general capabilities

The general capabilities encompass the knowledge, skills, behaviours and dispositions that will assist students to live and work successfully in the twenty-first century. Teachers may find opportunities to incorporate the capabilities into the teaching and learning program for Computer Science. The general capabilities are not assessed unless they are identified within the specified unit content.

Literacy

Students become literate as they develop the knowledge, skills and dispositions to interpret and use language confidently for learning and communicating in and out of school and for participating effectively in society. Literacy involves students listening to, reading, viewing, speaking, writing and creating oral, print, visual and digital texts, and using and modifying language for different purposes in a range of contexts.

In the Computer Science ATAR course, students develop literacy capability as they learn how to communicate ideas, concepts and detailed proposals to a variety of audiences; recognise how language can be used to manipulate meaning; and read and interpret detailed written instructions. They learn to understand and use language to discuss and communicate information, concepts and ideas related to the course.

By gaining literacy in the metalanguage of computer science, students understand that language varies according to context, and they increase their ability to use language flexibly. Computer science vocabulary is often technical and includes specific terms for concepts, processes and production. Students learn to understand that much technological information is presented in the form of drawings, diagrams, flow charts, models, tables and graphs. They also learn the importance of listening and talking when learning about technologies processes, especially in articulating, questioning and evaluating ideas.

Numeracy

Students become numerate as they develop the knowledge and skills to use mathematics confidently across other learning areas at school and in their lives more broadly. Numeracy involves students in recognising and understanding the role of mathematics in the world, and having the dispositions and capacities to use mathematical knowledge and skills purposefully.

In the Computer Science ATAR course, students work with the concepts of number, geometry, scale and proportion. They use models, create accurate technical drawings, work with digital models and use computational thinking in decision-making processes when designing and creating best-fit solutions.

Information and communication technology capability

Students develop information and communication technology (ICT) capability as they learn to use ICT effectively and appropriately to access, create and communicate information and ideas, solve problems and work collaboratively, and in their lives beyond school. The capability involves students in learning to make the most of the digital technologies available to them. They adapt to new ways of doing things as technologies evolve, and limit the risks to themselves and others in a digital environment.

In the Computer Science ATAR course, students create solutions that consider social and environmental factors when operating digital systems with digital information. They develop an understanding of the characteristics of data, digital systems, audiences, procedures and computational thinking. They apply this understanding when they investigate, communicate and create purpose-designed digital solutions. Students

learn to formulate problems, logically organise and analyse data and represent it in abstract forms. They automate solutions through algorithmic logic. Students decide the best combinations of data, procedures and human and physical resources to generate efficient and effective digital solutions.

Critical and creative thinking

Students develop capability in critical and creative thinking as they learn to generate and evaluate knowledge, clarify concepts and ideas, seek possibilities, consider alternatives and solve problems. Critical and creative thinking is integral to activities that require students to think broadly and deeply using skills, behaviours and dispositions, such as reason, logic, resourcefulness, imagination and innovation in all learning areas at school and in their lives beyond school.

In the Computer Science ATAR course, students develop capability in critical and creative thinking as they imagine, generate, develop, produce and critically evaluate ideas. They develop reasoning and the capacity for abstraction through challenging problems that do not have straightforward solutions. Students analyse problems, refine concepts and reflect on the decision-making process by engaging in systems, design and computational thinking. They identify, explore and clarify technologies, information and use that knowledge in a range of situations. Students think critically and creatively. They consider how data and information systems impact on our lives and how these elements might be better designed and managed.

Personal and social capability

Students develop personal and social capability as they learn to understand themselves and others, and manage their relationships, lives, work and learning more effectively. The capability involves students in a range of practices, including recognising and regulating emotions, developing empathy for others and understanding relationships, establishing and building positive relationships, making responsible decisions, working effectively in teams, handling challenging situations constructively and developing leadership skills.

In the Computer Science ATAR course, students develop personal and social capability as they engage in project management and development in a collaborative workspace. They direct their own learning, plan and carry out investigations, and become independent learners who can apply design thinking, technologies understanding and skills when making decisions. Students develop social and employability skills through working cooperatively in teams, sharing resources, tools, equipment and processes, making group decisions, resolving conflict and showing leadership. Designing and innovating involve a degree of risk-taking and as students work with the uncertainty of sharing new ideas they develop resilience.

The Computer Science ATAR course enhances students' personal and social capability by developing their social awareness. Students develop understanding of diversity by researching and identifying user needs. They develop social responsibility through the understanding of empathy with and respect for others.

Ethical understanding

Students develop ethical understanding as they identify and investigate concepts, values, character traits and principles, and understand how reasoning can help ethical judgement. Ethical understanding involves students in building a strong personal and socially oriented, ethical outlook that helps them to manage context, conflict and uncertainty, and to develop an awareness of the influence that their values and behaviour have on others.

In the Computer Science ATAR course, students develop the capacity to understand and apply ethical and socially responsible principles when collaborating with others and creating, sharing and using technologies,

data, processes, tools and equipment. In this course, students consider their own roles and responsibilities as discerning citizens, and learn to detect bias and inaccuracies. Understanding the protection of data, intellectual property and individual privacy in the school environment helps students to be ethical digital citizens.

Intercultural understanding

Students develop intercultural understanding as they learn to value their own cultures, languages and beliefs, and those of others. They come to understand how personal, group and national identities are shaped, and the variable and changing nature of culture. The capability involves students in learning about and engaging with diverse cultures in ways that recognise commonalities and differences, create connections with others and cultivate mutual respect.

In the Computer Science ATAR course, students consider how technologies are used in diverse communities at local, national, regional and global levels, including their impact and potential to transform people's lives. They explore ways in which past and present practices enable people to use technologies to interact with one another across cultural boundaries.

Representation of the cross-curriculum priorities

The cross-curriculum priorities address contemporary issues which students face in a globalised world. Teachers may find opportunities to incorporate the priorities into the teaching and learning program for the Computer Science ATAR course. The cross-curriculum priorities are not assessed unless they are identified within the specified unit content.

Aboriginal and Torres Strait Islander histories and cultures

The Computer Science ATAR course may provide opportunities for students to explore creative, engaging and diverse learning contexts for students to value and appreciate the contribution by the world's oldest continuous living cultures to past, present and emerging technologies.

Asia and Australia's engagement with Asia

The Computer Science ATAR course may provide opportunities for students to explore contemporary and emerging technological achievements that the Asia and Pacific regions have made, and continue to make, to global technological advances, including: innovation in hardware and software design and development; the regions' role in outsourcing of information and communications technology (ICT) services; and globalisation. Students could also consider the contribution of Australia's contemporary and emerging technological achievements to the Asia and Pacific Regions.

Sustainability

The Computer Science ATAR course may provide opportunities for students, within authentic contexts, to choose and evaluate digital technologies and information systems with regard to risks and opportunities they present. They may also evaluate the extent to which digital solutions can embrace and promote sustainable practices.

Unit 1 – Design and development of programming and networking solutions

Unit description

In this unit, students gain knowledge and skills to create software solutions. They use algorithms and structured programming to design and implement software solutions for a range of problems. They consider the complex interactions between users, developers, the law, ethics and society when computer systems are used and developed. Students learn about network communications and the transfer of data through a network.

A major focus of the course is the creation of systems and digital solutions to specific problems. In creating solutions, it is expected that students will use a structured development process to guide their approach. This development process is iterative in nature and involved four phases – investigating the problem, developing ideas and designing a solution, developing a solution and evaluating the effectiveness of the solution.

Unit content

This unit includes the knowledge, understandings and skills described below. This is the examinable content.

The unit content includes theoretical aspects (Knowledge) and practical aspects (Skills), and these are organised into two content areas:

- Programming
- Network communications.

Programming

Programming skills and concepts

Knowledge

- characters represented as numbers in binary, decimal and hexadecimal
- program control structures
  - sequence
  - selection
  - iteration
- modular coding using functions, parameters and arguments
  - scope of variables (Global, Local)
- data types used in solutions, including:
  - integer
  - float
  - string
  - Boolean

- types of operators
  - arithmetic operators (+, -, *, /, % or MOD)
  - relational operators (==,!=, >, <, >=, <=)
  - logical operators (AND, OR, NOT)
- identify the characteristics of the following data structures:
  - one-dimensional array

Skills

- apply, using pseudocode and a programming language, the following program control structures
  - sequence
  - selection
  - iteration
- use modular coding using functions, parameters and arguments
  - scope of variables (Global, Local)
- apply, using pseudocode and a programming language, data types used in solutions, including:
  - integer
  - float
  - string
  - Boolean
- use different types of operators
  - arithmetic operators (+, -, *, /, %)
  - relational operators (==, !=, >, <, >=, <=)
  - logical operators (AND, OR, NOT)
- read and write complex logical expressions including Boolean operators
  - AND, OR, NOT
  - logical order of precedence
- apply, using pseudocode and a programming language the following data structures:
  - one-dimensional array

Good programming practice

Knowledge

- Framework for development
  - investigate
    - problem description
    - define requirements
    - development schedule
  - design
    - design data structures
    - design and test algorithm
  - develop
    - develop and debug code
    - unit testing and use of live data

    - evaluate
        - user acceptance testing
        - developer retrospective
- good programming practice, including:
    - validate input before processing
    - use of meaningful variable names
    - use constants for readability and maintenance
    - use of comments to explain code
    - appropriate use of standard control structures
    - use of appropriate indentation and white space
    - one logical task per module
    - meaningful names for modules
    - exception handling

Skills

- apply the framework for development
- apply good programming practice, including:
    - validate input before processing
    - use of meaningful variable names
    - use of constants for readability and maintenance
    - use of comments to explain code
    - appropriate use of standard control structures
    - use of appropriate indentation and white space
    - one logical task per module
    - meaningful names for modules
    - exception handling

Structured algorithms

Knowledge

- benefits of using structured algorithms
    - ease of development
    - ease of understanding
    - ease of modification
- using pseudocode as a method for representing algorithms
- efficient algorithm design
    - use of a modular approach
    - structure charts as a design tool
    - use of stubs to represent incomplete modules

Skills

- using pseudocode to represent algorithms
- design efficient algorithms
    - use of a modular approach

  - structure charts as a design tool
  - use of stubs to represent incomplete modules

- use of standard algorithms
  - processing of arrays, including:
    - load an array and print its contents
    - add the contents of an array of numbers
    - identify position of minimum or maximum value
  - processing of sequential text files, including:
    - open for read, write and append
    - read and process data
    - write and append content
    - close

Testing

Knowledge

- appropriate test data, including:
  - data that test all the pathways through the algorithm
  - data that test boundary conditions ‘at’, ‘above’ and ‘below’ values upon which decisions are based
  - data where the required answer is known
  - type and range checking

Skills

- identify and select appropriate data to test an algorithm, including:
  - data that test all the pathways through the algorithm
  - data that test boundary conditions ‘at’, ‘above’ and ‘below’ values upon which decisions are based
  - data where the required answer is known
  - type and range checking
- testing both algorithms and coded solutions with test data, such as:
  - desk checking an algorithm (trace table)
  - stepping through a coded solution

Error detection and debugging code

Knowledge

- type of coding errors, including:
  - syntax error
  - runtime errors
  - logic errors

Skills

- debugging output statements
  - additional print statements in the code for use in the debugging process
    - used to identify which sections of the code have been executed
    - used to interrogate variable contents at a particular point in the execution of a program

External modules

Knowledge

- API (application programming interface)
  - purpose of an API
  - use of an API when developing software

Ethical and legal implications of software development

Knowledge

- concepts associated with piracy and copyright, including:
  - intellectual property
    - plagiarism in relation to the acknowledgement of code
    - Australian copyright laws
  - software licensing

Network Communications

Models of networking

Knowledge

- purpose of Department of Defense Transmission Control Protocol/Internet Protocol (DoD TCP/IP model)
- layers of DoD TCP/IP model
  - application
  - transport
  - internet
  - network
- role of layers within the model
- key protocols associated with layers
- role of IP addresses
- role of subnet masks
- key differences between IPv4 vs IPv6

Network components

Knowledge

- the function of networking components at different layers of TCP/IP model
  - transmission media (UTP, fibre optics, wireless)
  - router
  - switch
  - wireless access point
  - firewall

Network security

Knowledge

- need for preventing unauthorised access to a network
- role of firewalls in securing networks
- role of operating systems in network security

Network performance

Knowledge

- factors that affect network performance:
  - bandwidth
  - network design
  - data collisions
  - excess broadcast traffic

Skills

- create network diagrams using the CISCO network diagrammatic conventions to represent network topologies for LAN, WLAN and WAN

Unit 2 – Design and development of database solutions and cyber security considerations

Unit description

In this unit, students learn about the design concepts and tools used to develop relational database systems. Students gain skills to create database solutions and create queries to extract relevant information. Students consider the security of network communications, exploring a range of threats and measures used to keep networks secure Students examine attitudes and values of the creation and use of computer-based systems and their effect on society. They examine the ethical and legal obligations of the user and developer in the collection and storage of data.

This unit focuses on the creation of database systems. Students are expected to follow the technology process in order to produce quality products. The process involves four steps; investigate, design, produce and evaluate. This process is essential for the creation of solutions in the Computer Science course.

Unit content

This unit builds on the content covered in Unit 1.

This unit includes the knowledge, understandings and skills described below. This is the examinable content.

The unit content includes theoretical aspects (Knowledge) and practical aspects (Skills), and these are organised into two content areas:

- Cyber security
- Data management.

Cyber security

Ethics and Law

Knowledge

- role of ethical hacking in network security
  - purpose (improving security)
  - penetration testing
  - comparison with unethical hacking
- role of the P*rivacy Act 1988*
- the concept of the Australian privacy principles
- Australian Privacy Principles in relation to keeping data secure

Network security

Knowledge

- authentication
  - characteristics of strong passwords
  - organisational approach to password policies
  - password policies impact on data security
  - two-factor authentication
  - biometrics
- encryption
  - purpose of encryption
  - public vs private key encryption

Network threats

Knowledge

- distinguish between the different methods used to compromise the security of a system:
  - social engineering (phishing)
  - denial of service
  - back door
  - IP spoofing
  - SQL injection
  - man-in-the-middle
  - cross-site scripting
  - types of malware
  - physical security threats
  - zero day vulnerabilities

Cryptography

Knowledge

- purpose of cryptography
- plain text vs cipher text
- common ciphers
  - substitution
    - rotation cipher
    - random substitution
    - polyalphabetic cipher (e.g. Vigenère)
  - methods for cracking substitution ciphers
    - brute force
    - frequency analysis

Skills

- use common ciphers

Data management

Database management system (DBMS)

Knowledge

- relationship between data and information
- flat file vs relational database
- relational database management system (RDBMS)
  - role of a RDBMS in handling access to data
  - independence of data from RDBMS

Core database concepts

Knowledge

- organisation of a relational database
  - entities
  - attributes
  - relationships:
    - one-to-one
    - one-to-many
    - many-to-many
  - tables as the implementation of entities, consisting of fields and records
  - hierarchical structure of data
    - field/attribute
    - record
    - table/entity
  - datatypes
    - integer
    - float
    - Boolean
    - text
    - date
- primary and foreign keys to link tables
- composite key
- data anomalies
  - insert
  - update
  - delete

Data modelling

Knowledge

- purpose of database documentation for the developers
  - data dictionary
  - entity relationship (ER) diagrams using crow's foot notation

Skills

- analyse ER diagrams written in crow's foot notation (3 to 6 tables)
- create accurate ER diagrams (3 to 4 tables) using crow's foot notation
- create a data dictionary
- resolve many to many (M:N) relationship

Data integrity

Knowledge

- factors influencing integrity of data, including:
  - currency
  - authenticity
  - relevance
  - accuracy
  - outliers (cleaning)
- relationship between validity and accuracy of data

Normalisation

Knowledge

- purpose of normalising data to third normal form (3NF)
- know the process to normalise data to 3NF

Skills

- apply the process to normalise data to 3NF (three to four tables)
  - normalise data to 1NF
  - normalise data to 2NF
  - normalise data to 3NF

Database creation and manipulation

Skills

- use a RDBMS to create and manipulate a relational database with a minimum of three tables
- use SQL to manipulate a database including:
  - SELECT
  - INSERT
  - DELETE

- UPDATE
- ORDER BY
- inner join across two tables
- aggregate functions (COUNT, SUM, AVG, MAX, MIN)

Development issues

Knowledge

- Ethical issues
  - collecting data about individuals
  - privacy concerns
  - appropriate use of data
  - Australian Privacy Principles applicable to the use of personally identifiable and sensitive data
- Security issues
  - keeping personal data private
  - backups of organisational data
  - restricting access to data
- Legal issues
  - implications of the *Privacy Act 1988* for developers

Assessment

Assessment is an integral part of teaching and learning that at the senior secondary years:

- provides evidence of student achievement
- identifies opportunities for further learning
- connects to the standards described for the course
- contributes to the recognition of student achievement.

Assessment for learning (formative) and assessment of learning (summative) enable teachers to gather evidence to support students and make judgements about student achievement. These are not necessarily discrete approaches and may be used individually or together, and formally or informally.

Formative assessment involves a range of informal and formal assessment procedures used by teachers during the learning process in order to improve student achievement and to guide teaching and learning activities. It often involves qualitative feedback (rather than scores) for both students and teachers, which focuses on the details of specific knowledge and skills that are being learnt.

Summative assessment involves assessment procedures that aim to determine students' learning at a particular time, for example when reporting against the standards, after completion of a unit/s. These assessments should be limited in number and made clear to students through the assessment outline.

Appropriate assessment of student work in this course is underpinned by reference to the set of pre-determined course standards. These standards describe the level of achievement required to achieve each grade, from A to E. Teachers use these standards to determine how well a student has demonstrated their learning.

Where relevant, higher order cognitive skills (e.g. application, analysis, evaluation and synthesis) and the general capabilities should be included in the assessment of student achievement in this course. All assessment should be consistent with the requirements identified in the course assessment table.

Assessment should not generate workload and/or stress that, under fair and reasonable circumstances, would unduly diminish the performance of students.

School-based assessment

The *Western Australian Certificate of Education (WACE) Manual* contains essential information on principles, policies and procedures for school-based assessment that must be read in conjunction with this syllabus.

School-based assessment involves teachers gathering, describing and quantifying information about student achievement.

Teachers design school-based assessment tasks to meet the needs of students. As outlined in the *WACE Manual*, school-based assessment of student achievement in this course must be based on the Principles of Assessment:

- Assessment is an integral part of teaching and learning
- Assessment should be educative
- Assessment should be fair

- Assessment should be designed to meet its specific purpose/s
- Assessment should lead to informative reporting
- Assessment should lead to school-wide evaluation processes
- Assessment should provide significant data for improvement of teaching practices.

Summative assessments in this course must:

- be limited in number to no more than eight tasks
- allow for the assessment of each assessment type at least once over the year/pair of units
- have a minimum value of 5 per cent weighting of the total school assessment mark
- provide a representative sampling of the syllabus content.

Assessment tasks not administered under test or controlled conditions require appropriate authentication processes.

Assessment table – Year 11

Type of assessment	Weighting
Project The student is required to develop a database and/or software solution by using a structured development process. Students use the skills and theory covered in this course to create working solutions based on provided stimulus materials. Software solutions should make use of modular programming techniques. Database solutions should make use of SQL to create and manipulate a relational database system. Stimulus material can include: diagrams; extracts from newspapers and/or journal articles; data dictionaries; structure charts; trace tables; algorithms and/or algorithm segments (in pseudocode); and/or screen captures or representations of databases and programs. Diagrams can include: entity relationship diagrams, computer system diagrams, and/or network diagrams. The student is required to research ideas; implement a database and/or software system using a relational database management system and/or programming language; explore, develop and evaluate solutions; and manage processes throughout production to produce solutions. Projects should be open-ended to give students the opportunity to apply a structured development process. The project should require students to investigate and deconstruct a problem, design a suitable solution to the problem, develop a solution based on their design and evaluate the effectiveness of their final product. Teachers are encouraged to interrelate topics such as programming and databases where possible to create authentic problems requiring a solution.	40%

Type of assessment	Weighting
Theory test Tests typically consist of a combination of questions requiring short and extended answers. Short answer questions can be a mix of closed and open items that can be scaffolded or sectionalised. The student can be required to: explain concepts, apply knowledge, analyse and/or interpret data and/or refer to stimulus material. Stimulus material can include: diagrams; extracts from newspaper and/or journal articles; data dictionaries; structure charts; trace tables; algorithms and/or algorithm segments (in pseudocode); and/or screen captures or representations of databases and programs. Diagrams can include: entity relationship diagrams and/or network diagrams. Extended answer questions can be a mix of closed and open items that can be scaffolded or sectionalised, typically with an increasing level of complexity. The student can be required to apply knowledge and/or critical thinking skills; analyse and/or interpret data, extended algorithms, relational databases, tables and/or diagrams; devise labelled diagrams, and/or solutions (or parts of solutions). Some questions can require the student to refer to stimulus material. Stimulus material can include: diagrams; extracts from newspaper and/or journal articles; data dictionaries; structure charts; trace tables; algorithms and/or algorithm segments (in pseudocode); and/or screen captures or representations of databases and programs. Diagrams can include: entity relationship diagrams and/or network diagrams.	20%
Practical test Tests typically consist of a set of questions developing and implementing solutions requiring the use of a programming language and/or a relational database management system to meet specified requirements. Programming skills assessed could include: writing code; and/or compiling, testing and/or debugging program code. Database skills assessed could include: creating and implementing schema, joining tables and ordering using structured query language.	10%
Examination Typically conducted at the end of each semester and/or unit and reflecting the examination design brief for this syllabus.	30%

Teachers must use the assessment table to develop an assessment outline for the pair of units (or for a single unit where only one is being studied).

The assessment outline must:

- include a set of assessment tasks
- include a general description of each task
- indicate the unit content to be assessed
- indicate a weighting for each task and each assessment type

- include the approximate timing of each task (for example, the week the task is conducted, or the issue and submission dates for an extended task).

In the assessment outline for the pair of units, each assessment type must be included at least over the year/pair of units.

The set of assessment tasks must provide a representative sampling of the content for Unit 1 and Unit 2.

Assessment tasks not administered under test/controlled conditions require appropriate validation/authentication processes.

Reporting

Schools report student achievement, underpinned by a set of pre-determined standards in terms of the following grades:

Grade	Interpretation
A	Excellent achievement
B	High achievement
C	Satisfactory achievement
D	Limited achievement
E	Very low achievement

The grade descriptions for the Computer Science ATAR Year 11 syllabus are provided in Appendix 1. They are used to support the allocation of a grade. They can also be accessed, together with annotated work samples, on the course page of the Authority website at www.scsa.wa.edu.au.

To be assigned a grade, a student must have had the opportunity to complete the education program, including the assessment program (unless the school accepts that there are exceptional and justifiable circumstances).

Refer to the *WACE Manual* for further information about the use of a ranked list in the process of assigning grades.

The grade is determined by reference to the standard, not allocated on the basis of a pre-determined range of marks (cut-offs).

Appendix 1 – Grade descriptions Year 11 (provisional)

A	**Knowledge and understanding** Accurately uses computer science terminology and describes processes and concepts in context and with justification.
	Cyber security and networking Accurately explains network threats and security solutions. Successfully designs and justifies network diagrams applying correct and appropriate conventions, connections and topology. Justifies network design decisions and demonstrates a consideration of network performance issues. Consistently and accurately interprets and analyses network diagrams.
	Data management skills Consistently and accurately constructs appropriate entity relationship diagrams reflecting system requirements, including relevant use of diagrammatic conventions, relationships, cardinality, attributes, and primary and foreign keys. Consistently and accurately interprets and analyses entity relationship diagrams. Consistently applies normalisation to the third normal form to accurately model a simple database solution. Consistently and accurately constructs appropriate multi-table relational databases reflecting system requirements and uses a range of appropriate query techniques to extract relevant data.
	Programming skills Consistently applies suitable methodology accurately to assist the software development process. Consistently designs and creates pseudocode with appropriate use of standards and conventions. Consistently and accurately applies programming control structures in pseudocode and a programming language to develop efficient solutions that reflect software requirements. Consistently uses a range of appropriate data structures, including arrays, to develop effective software solutions. Consistently uses effective modular programming techniques, including parameter passing, to develop working software solutions that meet requirements. Uses a broad range of techniques to effectively test and debug software solutions using appropriate test data. Consistently and accurately implements data validation and desk checking techniques. Consistently and accurately explains ethical and legal issues related to software development.

B	**Knowledge and understanding** Accurately uses computer science terminology and describes processes and concepts in context.
	Cyber security and networking Describes network threats and security solutions. Successfully designs network diagrams applying correct and appropriate conventions, connections and topology. Explains network design decisions and demonstrates a consideration of network performance issues. Interprets and analyses network diagrams.
	Data management skills Constructs appropriate entity relationship diagrams reflecting system requirements, including relevant use of diagrammatic conventions, relationships, cardinality, attributes and primary and foreign keys. Interprets and explains entity relationship diagrams. Applies normalisation to the third normal form to accurately model a simple database solution. Constructs appropriate multi-table relational databases accurately reflecting system requirements and using appropriate queries to extract relevant data.
	P**rogramming skills** Consistently applies suitable methodology to assist the software development process. Designs and creates pseudocode with appropriate use of standards and conventions. Applies programming control structures in pseudocode and a programming language to develop efficient solutions that reflect software requirements. Uses a range of data structures, including arrays, to develop software solutions. Uses modular programming techniques, including parameter passing, to develop working software solutions that meet requirements. Uses a range of techniques to effectively test and debug software solutions using appropriate test data. Implements data validation and desk checking techniques. Consistently describes ethical and legal issues related to software development.

C	**Knowledge and understanding** Uses computer science terminology and describes processes and concepts.
	Cyber security and networking Identifies network threats and security solutions. Designs limited network diagrams using conventions, showing connections between devices. Identifies network design decisions that have an effect on performance issues. Interprets network diagrams with varying success.
	Data management skills Constructs entity relationship diagrams, with errors, that attempt to reflect system requirements. Entity relationship diagrams use diagrammatic conventions, relationships, cardinality, attributes, and primary and foreign keys. Interprets entity relationship diagrams. Attempts to apply normalisation to third normal form to model a database solution. Constructs a multi-table relational database, reflecting system requirements and using simple queries to extract data.
	Programming skills Applies suitable methodology to software development. Designs and creates pseudocode using appropriate standards and conventions, with some errors. Applies programming control structures in pseudocode and programming languages, which may include minor errors in logic and conventions but partially reflects software requirements. Uses a limited range of data types and structures, such as arrays to develop software solutions. Uses modular programming techniques, including parameter passing, to develop part of a software solution. Is able to test software design and solution, including selection of test data to identify errors. Attempts to implement data validation and desk checking techniques. Identifies ethical and legal issues related to software development.

D	**Knowledge and understanding** Attempts to use computer science terminology to describe processes and concepts.
	Cyber security and networking Attempts to design limited network diagrams but uses with incorrect conventions and logic.
	Data management skills Constructs inaccurate entity relationship diagrams with significant logic and convention errors. Inaccurately interprets entity relationship diagrams. Constructs an incomplete database with limited functionality.
	Programming skills Applies unsuitable methodology to software development. Represents incomplete algorithms using pseudocode with standard and convention errors. Develops algorithms and programs which are incomplete and inaccurate, demonstrating little understanding of programming control structures. Develops incomplete software solutions. Attempts, or presents, unfinished modular programming techniques. Creates trace tables which are incomplete, inaccurate or unable to test algorithmic logic. Creates test data which is incomplete, inaccurate or unable to test program functionality. Incorrectly identifies ethical and legal issues related to software development.

E	Does not meet the requirements of a D grade and/or has completed insufficient assessment tasks to be assigned a higher grade.

Note: these grade descriptions will be reviewed at the end of the second year of implementation of this syllabus.