Sample Course Outline

Computer Science

ATAR Year 11

**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.

**Copyright**

© School Curriculum and Standards Authority, 2023

This document – apart from any third-party copyright material contained in it – may be freely copied, or communicated on an intranet, for non-commercial purposes in educational institutions, provided that the School Curriculum and Standards Authority (the Authority) is acknowledged as the copyright owner, and that the Authority’s moral rights are not infringed.

Copying or communication for any other purpose can be done only within the terms of the *Copyright Act 1968* or with prior written permission of the Authority. Copying or communication of any third-party copyright material can be done only within the terms of the *Copyright Act 1968* or with permission of the copyright owners.

Any content in this document that has been derived from the Australian Curriculum may be used under the terms of the [Creative Commons Attribution 4.0 International licence](http://creativecommons.org/licenses/by/4.0/).

**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.

Sample course outline

Computer Science – ATAR Year 11

Semester 1 – Unit 1 (Design and development of programming and network solutions)

| **Week** | **Syllabus content** | |
| --- | --- | --- |
| **Knowledge** | **Skills** |
| 1–3 | **Introduction**   * overview of Semester 1 * assessment requirements   **Programming**  **Programming skills and concepts**   * characters represented as numbers in binary, decimal and hexadecimal * program control structures   + sequence   + selection   + iteration * modular coding using functions   + scope of variables (Global, Local)   + parameters and arguments * data types used in solutions, including:   + integer   + float   + string   + Boolean * types of operators   + arithmetic operators (+, -, \*, /, % or MOD)   + relational operators (==, !=, >, <, >=, <=)   + logical operators (AND, OR, NOT) | **Programming**  **Programming skills and concepts**   * apply, using pseudocode and a programming language, the following program control structures   + sequence   + selection   + iteration * use modular coding using functions   + scope of variables (Global, Local)   + parameters and arguments * 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) |
| 4 | * identify the characteristics of the following data structures:   + one-dimensional array | * 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 |
| 5–6 | **Good programming practice**   * Framework for development   + investigate     - problem description     - define requirements     - development schedules, including Gantt charts   + design     - design data structures     - design and test algorithm   + develop     - develop and debug code     - unit testing   + 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 | **Good programming practice**   * 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 |
| 7–8 | **Structured algorithms**   * 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 | **Structured algorithms**   * 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 |
| 9–10 | **Testing**   * appropriate test data, including:   + data that tests all the pathways through the algorithm   + data that tests boundary conditions ‘at’, ‘above’ and ‘below’ values upon which decisions are based   + data where the required answer is known   + type and range checking   **Error detection and debugging code**   * type of coding errors, including:   + syntax error   + runtime errors   + logic errors   **Ethical and legal implications of software development**   * concepts associated with piracy and copyright, including:   + intellectual property     - plagiarism in relation to the acknowledgement of code     - Australian copyright laws     - purpose of software licensing     - open source     - proprietary   **External modules**   * API (application programming interface)   + purpose of an API   + use of an API when developing software | **Testing**   * identify and select appropriate data to test an algorithm, including:   + data that tests all the pathways through the algorithm   + data that tests 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**   * 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 |
| 11–12 | Network Communications **Models of networking**   * 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**   * the function of networking components at different layers of TCP/IP model   + transmission media (UTP, fibre optics, wireless)   + modem   + router   + switch   + wireless access point   + firewall   **Network security**   * need for preventing unauthorised access to a network * role of firewalls in securing networks * role of operating systems in network security |  |
| 13–14 | **Network performance**   * factors that affect network performance:   + bandwidth   + network design   + data collisions   + excess broadcast traffic | **Network performance**   * create logical network diagrams using the CISCO network diagrammatic conventions to represent network topologies for LAN, WLAN and WAN |
| 15 | Revision | |
| 16 | Semester 1 examination | |

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

| **Week** | **Syllabus content** | |
| --- | --- | --- |
| **Knowledge** | **Skills** |
| 1–2 | Course review  * review Unit 1 * review assessment requirements * overview of Semester 2  Cyber security **Ethics and Law**   * role of ethical hacking in network security   + purpose (improving security)   + penetration testing   + comparison with unethical hacking * role of the *Privacy Act 1988* * the concept of the *Australian Privacy Principles* * *Australian Privacy Principles* in relation to keeping data secure   **Network security**   * 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 key (asymmetric) encryption   + symmetric encryption |  |
| 3–4 | **Network threats**   * 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     - viruses     - worms     - Trojan horses     - spyware     - adware     - ransomware   + physical security threats   + zero-day vulnerabilities   **Security frameworks**   * the CIA Triad model of security analysis   + Confidentiality   + Integrity   + Availability * the AAA framework for securing systems   + Authentication   + Authorisation   + Accounting   **Cryptography**   * 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 | **Security frameworks**   * use the CIA Triad to analyse security threats and incidents * use the AAA framework for security analysis and auditing   **Cryptography**   * use common ciphers |
| 5–8 | **Data management**  **Database management system (DBMS)**   * relationship between data and information * flat file vs relational database * relational database managements system (RDBMS)   + role of a RDBMS in handling access to data   + independence of data from RDBMS   **Core database concepts**   * 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**   * purpose of database documentation for the developers   + data dictionary   + entity relationship (ER) diagrams using crow’s foot notation (see support document)   **Data integrity**   * factors influencing integrity of data, including:   + currency   + authenticity   + relevance   + accuracy   + outliers (cleaning) * relationship between validity and accuracy of data | **Data modelling**   * analyse ER diagrams written in crow’s foot notation (3 to 6 tables) * create accurate ER diagrams (minimum of four tables) using crow’s foot notation * create a data dictionary (see support document) * resolve many-to-many (M:N) relationship |
| 9–11 | **Normalisation**   * purpose of normalising data to third normal form (3NF) * know the process to normalise data to 3NF | **Normalisation**   * apply the process to normalise data to 3NF (minimum of 4 tables)   + normalise data to 1NF   + normalise data to 2NF   + normalise data to 3NF |
| 12–13 |  | **Database creation and manipulation**   * 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 joins   + aggregate functions (COUNT, SUM, AVG, MAX, MIN) |
| 14 | **Development issues**   * 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 |  |
| 15 | Revision | |
| 16 | Semester 2 examination | |

Acknowledgements

**Semester 1 – Unit 1 (Design and development of programming and network solutions)**

9–10 NSW Stage 6 Software Design and Development Syllabus (1999).

© 1999 NSW Education Standards Authority (NESA) for and on behalf of the Crown in right of the State of New South Wales