Computer Science

LEVEL 3	15 TCE CREDIT POINTS
COURSE CODE	ITC315118
COURSE SPAN	2018 — 2023
READING AND WRITING STANDARD	NO
MATHEMATICS STANDARD	NO
COMPUTERS AND INTERNET STANDARD	YES

This course is current for 2022.

Computer Science involves the study of the processes underlying the storage, transformation and transfer of data

It includes both the theoretical study of algorithms and the practical problems involved in implementing them, usually via a programming language.

Rationale

Computer Science involves the study of the processes underlying the storage, transformation and transfer of data. It includes both the theoretical study of algorithms and the practical problems involved in implementing them, usually via a programming language.

Computer Science can be a starting point for further education in Information and Communications Technology (ICT) or engineering or a preparation for the vast range of careers that require efficient and effective use of ICT and computational thinking. Predicted ICT skills shortages, both within Australia and globally, suggest that professionals with a computer science background will be in demand. In addition, computer science skills are a major driver of economic growth and productivity (ACS/DAE, Australia's Digital Pulse Developing the digital workforce to drive growth in the future 2016, Australian Computer Society/Deloitte Access Economics, Sydney, NSW).

Aims

Computer Science aims to develop learners':

- ability to identify, analyse and design algorithms (sets of instructions to complete a task)
- understanding of programming concepts in a formal computer programming language
- skill in the design, documentation, analysis, testing and evaluation of computer programs
- understanding of Boolean logic, binary data, representation of data types and their relationship to underlying digital hardware
- familiarity with high and low-level programming languages
- awareness of the social, ethical and professional aspects of computer science.

Learning Outcomes

On successful completion of this course, learners will be able to:

- apply the techniques of computer science to a range of problems and appreciate limitations of using algorithmic solutions
- design, document, compare, evaluate and refine algorithmic and programming solutions to a range of problems expressed in a variety of forms
- describe software and hardware aspects of computing, and explain their operation with the underpinning mathematics and science of the discipline
- describe interaction between people and computers, and implications for software design
- identify career and further education opportunities that make use of computer science skills, knowledge and understanding
- describe societal consequences of technological solutions and the professional and ethical responsibility of people working within this field
- manage their own learning, including time management and organisation skills
- develop Java applications.

Access

Learners need to be able to operate a computer in order to complete a range of computer programming exercises, and access documentation and other relevant material on the internet.

Pathways

It is expected that learners entering this course would have well-developed ICT, numeracy and literacy skills. Experience in problem solving, including logical and critical thinking, would be advantageous.

Learners wishing to pursue a computing career could use this as a starting point to study a degree at University, or VET Certificate IV, or Diploma, including combined Diploma/Degree courses. These courses may focus on multimedia and the internet, artificial intelligence, mobile and ubiquitous computing, systems and networks, computer security, distributed systems, software engineering or programming languages. Learners entering the workforce should expect to undergo further education and training.

Recent research shows many people in ICT careers are working in industries outside of ICT itself (ACS/DAE 2016, p. 3). An increasing number of careers involve computer science, and students may find this course useful in other fields such as: law; medical research; engineering; logistics; military; tourism; commerce, and management.

Resource Requirements

All learners will require access to computers with the following minimum requirements:

- access to the internet, including web and email
- access to a computer on which they can install and run applications
- Java Development Kit and an appropriate environment for creating, editing and running programs
- word processing facilities
- cloud or local file storage
- printing.

Course Size And Complexity

This course has a complexity level of 3.

At Level 3, the learner is expected to acquire a combination of theoretical and/or technical and factual knowledge and skills and use judgment when varying procedures to deal with unusual or unexpected aspects that may arise. Some skills in organising self and others are expected. Level 3 is a standard suitable to prepare learners for further study at tertiary level. VET competencies at this level are often those characteristic of an AQF Certificate III.

This course has a size value of 15.

Course Description

The course consists of three major content areas plus a Computing Option which relies on knowledge from those content areas.

AREA 1: PROBLEM SOLVING AND PROGRAMMING
This content area includes algorithm design and problem solving, efficiency and reliability, the Software Development Life Cycle, principles of programming practice and style including documentation, Object-oriented programming in the Java language and Human Computer Interaction and interface design.

AREA 2: COMPUTING FUNDAMENTALS AND COMPUTER LIMITATIONS
This content area includes representation of data at machine level, and data types, formal logic and logic laws and principles of machine code.

AREA 3: SOCIAL/ETHICAL ISSUES AND PROFESSIONAL RESPONSIBILITY
This content area include social and ethical issues associated with the computing profession and career pathways and the role of professional bodies.

AREA 4: COMPUTING OPTION
In the Computing Option learners pursue an area of interest in more depth. This can include development of a software application to meet a client need, or application of CS principles to a new programming environment or tool.

Course Requirements

All content areas of Computer Science are compulsory. While the order of study is not prescribed, learners will negotiate with providers to ensure that relevant sections of the three major content areas (Problem Solving and Programming, Computer Fundamentals and Computer Limitations, and Social/Ethical Issues and Professional Responsibility) have been undertaken prior to beginning the Computing Option. It is not envisaged that the following components be taught as isolated topics but rather as an integrated body of knowledge.

All listed content is compulsory.

Course Content

AREA 1: PROBLEM SOLVING AND PROGRAMMING (70 HOURS)

Algorithms and programming solutions to a variety of problems are designed and expressed in a variety of forms. Students will develop skills in understanding the problem, exploring problem solving strategies, design and creation of a solution. Algorithms that require mathematical solutions, such as those involving summation and searching, are investigated.

A fundamental understanding of the Software Development Life Cycle (design, code, test, evaluate and refine) is required. Practical activities need to provide experience for learners in all stages of this cycle and to develop an understanding of the importance of analysis and design before beginning to code. Programs should adhere to established programming styles and significant programs should have formal documentation.

ALGORITHM DESIGN AND PROBLEM SOLVING

This includes

- exploration of a range of problems - some problems cannot be reduced to an algorithm as they have no clear rules, some problems have clear rules but are difficult to put into algorithms because they have many possible responses to rules (e.g. the game of Go). Other problems are those for which rules are still to be discovered such as natural language processing and how to cure certain diseases. The course focuses on problems that can be represented algorithmically. This section should provide a link between Computer Science and other disciplines
- exploration of forms of problem solving – algebraic, algorithmic, trial and error
- the specific requirements of a problem are determined from problem definitions given in a variety of forms
- examination of different solutions to the same problem (critical evaluation, most efficient, reliable)
- problem deconstruction/decomposition – problems need to be broken down into a number of well-defined steps
- visualisation of solution – use of diagrams to illustrate the solution
- expression of the solution to the problem – use an initially/when model for event-driven solutions.

PROGRAMMING

Learners learn the fundamentals of Object-oriented programming and event driven programming.

Learners write a variety of Java applications using the following Java features

- primitive types (promotion and casting)
- arithmetic and logic operators (+, -, *, / , %, &&, !, ||), order of operations, and some mathematical functions -such as (Math.pow(), Math.random())
- classes and information hiding
- graphics (drawing, filling)
- pre-defined objects, including arrays and strings, and modifying and creating objects
- control-flow (selection using if else/switch, iteration using for/while)
- methods, parameters and scope
- GUIs (widgets including buttons, text fields, labels)
- events and listeners using the AWT library.

A key component of programming will be an emphasis on good programming practice. Programs need to adhere to a defined set of standards including good variable name choice, commenting, and indenting. Learners will be introduced to examples of programming style guides, and the reasons why organisations often require their programmers to comply with a specific style.

A key element of this topic is designing applications for genuine solutions. Fundamental notions of HCI (Human Computer Interaction) are introduced.

TESTING AND EVALUATION

In this section

- a structured approach to testing is followed. Testing plans are written from program specifications in the absence of a program
- tracing as a means of debugging programs is introduced, including both hand and automated tracing
- self-review, peer review and external review (including by end users) is used to evaluate applets and applications and identify future refinements.

DOCUMENTATION

Programs of significant size should be accompanied by both technical and user documentation. Technical documentation includes internal comments of programs. User documentation provides a description of the program's purpose, operating instruction and appropriate online help.

Programming style guides could include (but are not limited to) the original Oracle Java Code Conventions, the Google Java Style Guide and others such as the JavaRanch Style Guide.

The general value of programming style is conveyed in publications such as Stanford's introductory programming style guide and the Harvard CS50 Style Guide (which is not designed for Java but outlines the reasons for style decisions). Documentation can take different forms but it is recommended that, where appropriate, technical documentation within programs be undertaken in accordance with the Javadoc "documentation comments" format.

AREA 2: COMPUTER FUNDAMENTALS AND COMPUTER LIMITATIONS (40 HOURS)

In order to come to an understanding of the limitations and possibilities for the use of computer technology into the future, learners need to understand computer architectures, and the role of the operating system.

Areas to be covered:

- binary number system for whole number and fraction and conversions to decimal and hexadecimal
- basic binary arithmetic (addition only)
- two's complement representation and arithmetic (addition and subtraction only)
- representation of primitive data types (integer, char, boolean, float)
- representation of non-numeric data using hexadecimal where appropriate (e.g. characters, colours, instructions)
- implications of representation of floating point numbers for accuracy of calculations
- representation of arrays as well as sound and picture files
- Boolean operators (AND, OR, NOT)
- logic gates, basic computer circuits and the flip-flop
- using truth tables, Karnaugh maps and simplifications using the specified list of logic laws to design logic circuits
- computer architecture – the fundamental components of a computer in the von Neumann architecture and the relevant historical context
- machine code and its relationship to high level languages such as Java
- the machine cycle required to add two numbers (fetch, decode, execute)
- operating systems and the role of the JVM
- newer technologies and their relationship to basic computer architecture.

AREA 3: SOCIAL / ETHICAL ISSUES AND PROFESSIONAL RESPONSIBILITY (10 HOURS)

Computer professionals have specialised knowledge and often have positions with authority. For this reason, they may have a significant impact on society. There is a duty to exercise that power responsibly.

Areas to be covered:

- career pathways, skills and education required
- the role of professional associations and codes of ethics
- responsibilities of the computing professional in the workplace
- responsibilities of those in positions of authority
- examples and consequences of technological errors, such as software bugs
- the consequences of good and bad user interface design, and the obligation to design for all users (for example, http://www.australia.gov.au/accessibility.)

AREA 4: COMPUTING OPTION (30 HOURS)

The skills gained in Computer Science are used to explore an area of interest in more depth. The option chosen must enable learners to demonstrate problem solving skills, research, and technical communication skills. In addition, learners must adhere to ethical and professional standards as they are prescribed in the course. The option product will be used to assess both Criteria 8 and 9 and either Criterion 1 or 6, along with at least one of the remaining criteria.

Providers may wish to direct the nature of the Computing Option based on the interests, needs and skills of individual learners.

Suggested topics include (but are not limited to):

- production of a Java application for a client following the software development lifecycle
- Object-oriented programming in other languages
- game development in a suitable environment
- exploration of network programming
- programming for mobile devices
- Media Computation
- cryptography, compression and security
- artificial intelligence and machine learning
- human-computer interaction
- computer forensics

- ethical and legal aspects of computer science
- application of computer science principles to another field (e.g. life sciences, psychology, law)
- big data and data science
- exploration of Java libraries
- exploration of alternative Java programming environments (e.g. Greenfoot, Robocode)
- programming LEGO® robots using LeJOS
- programming of embedded systems and microcontrollers
- digital electronics.

Note that the topic chosen must provide the learner with opportunity for the assessment of the Criteria stated above.

Work Requirements

AREA 1: WORK REQUIREMENTS

The equivalent of:

- Assessment tasks (10 programming tasks). Program development with accompanying explanation of specific aspects of the Java language.
- Assessment task – Algorithms (approximately 2 hours)
- Assessment task - SDLC (approximately 2 hours)

Fully documented programs (2 programs). These may form part of the Java language assessment tasks.

AREA 2: WORK REQUIREMENTS

The equivalent of:

- Assessment task - Logic laws, truth tables and Karnaugh maps (approximately 2 hours)
- Assessment task - Number representation(approximately 2 hours)
- Assessment task - Data representation (approximately 2 hours)
- Assessment task - TOY machine (approximately 2 hours)

These assessment tasks would typically include one or more problems to be solved, along with explanation of some aspect of the topic.

AREA 3: WORK REQUIREMENTS

The equivalent of:

- Assessment task - Social issues, professionalism and ethics (1000 words)

AREA 4: WORK REQUIREMENTS

Option Product: documented program, or report on investigation (4000 words or equivalent).

Assessment

Criterion-based assessment is a form of outcomes assessment that identifies the extent of learner achievement at an appropriate end-point of study. Although assessment – as part of the learning program – is continuous, much of it is formative, and is done to help learners identify what they need to do to attain the maximum benefit from their study of the course. Therefore, assessment for summative reporting to TASC will focus on what both teacher and learner understand to reflect end-point achievement.

The standard of achievement each learner attains on each criterion is recorded as a rating 'A', 'B', or 'C', according to the outcomes specified in the standards section of the course.

A 't' notation must be used where a learner demonstrates any achievement against a criterion less than the standard specified for the 'C' rating.

A 'z' notation is to be used where a learner provides no evidence of achievement at all.

Providers offering this course must participate in quality assurance processes specified by TASC to ensure provider validity and comparability of standards across all awards. For further information, see TASC's quality assurance and assessment processes.

Internal assessment of all criteria will be made by the provider. Providers will report the learner's rating for each criterion to TASC.

TASC will supervise the external assessment of designated criteria which will be indicated by an asterisk (*). The ratings obtained from the external assessments will be used in addition to internal ratings from the provider to determine the final award.

Quality Assurance Process

The following processes will be facilitated by TASC to ensure there is:

- a match between the standards of achievement specified in the course and the skills and knowledge demonstrated by learners
- community confidence in the integrity and meaning of the qualification.

TASC gives course providers feedback about any systematic differences in the relationship of their internal and external assessments and, where appropriate, seeks further evidence through audit and requires corrective action in the future.

External Assessment Requirements

The external assessment for this course will comprise:

- a written examination assessing criteria: 1, 2, 3, 4 and 5.

For further information see the current external assessment specifications and guidelines for this course available in the Supporting Documents below.

Criteria

The assessment for Computer Science Level 3 will be based on the degree to which the learner can:

* = denotes criteria that are both internally and externally assessed

1. design, extend and improve algorithmic solutions to a range of problems*
2. create programs in a high level programming language*
3. use appropriate objects in the design of programs*
4. describe and apply knowledge of computer architecture*
5. analyse how data are represented and stored*
6. apply the software development life cycle to a variety of problems
7. analyse and apply societal, professional and ethical responsibilities
8. apply personal skills to plan, organise and complete activities
9. communicate technological information

Standards

Criterion 1: design, extend and improve algorithmic solutions to a range of problems

This criterion is both internally and externally assessed.

Rating 'A'
In addition to the standards for a 'C' and a 'B' rating, the learner:

Rating 'B'
In addition to the standards for a 'C' rating, the learner:

Rating 'C'
The learner:

Rating A	Rating B	Rating C
explains the logic of complex algorithms and predicts their output or behaviour under a full range of input conditions	explains the logic of complex algorithms and predicts their output or behaviour for a given set of data	explains the logic of simple algorithms and predicts their output or behaviour for a given set of data
creates complete, complex algorithms from specifications	extends a section of an existing algorithm to add new required functionality	modifies a small section of an existing algorithm to meet stated requirements
when provided with a broad description of a required applications, develops a comprehensive algorithm for implementing that application and describes the application's interface	when provided with a faulty or incomplete section of a complex algorithm, identifies any missing events or components and uses given advice to repair or complete the algorithm	when provided with a faulty or incomplete section of a simple algorithm, uses given advice to repair or complete the algorithm
implements detailed, efficient and well-structured algorithms with multiple complex combinations of basic programming constructs. The algorithms include features such as sophisticated input validation and exception handling	creates algorithms that include complex combinations of basic programming constructs such as sequence, selection and iteration	creates algorithms for simple situations that include basic programming constructs such as sequence, selection and iteration
applies the appropriate design theory to a given problem, explores a range of possible solutions, evaluating each, and determines the best solution for an unfamiliar problem. The best solution is expressed in terms of an algorithm.	given an unfamiliar computing task, applies top-down design in order to identify the required subtasks and develops these into an algorithm for the task.	develops a valid algorithm as a solution to an unfamiliar computing task in which subtasks have been identified.

Criterion 2: create programs in a high level programming language

This criterion is both internally and externally assessed.

Rating 'A'
In addition to the standards for a 'C' and a 'B' rating, the learner:

Rating 'B'
In addition to the standards for a 'C' rating, the learner:

Rating 'C'
The learner:

Rating A	Rating B	Rating C
implements all required language constructs and	makes informed and reasoned	makes informed statements

interactions between components by writing and analysing Java code, and includes methods, parameters and scope of variables	statements about the outcome of major features of the Java language	about outcome of commonly used features of the Java language
traces sections of programs, designs suitable trace tables and uses appropriate and informative test data (for example, critical user input values)	traces sections of programs and designs suitable trace tables	traces sections of programs using specified values
analyses complex and extended sections of code to determine their purpose	uses appropriate strategies to determine the purpose of sections of code	examines small sections of programs to determine their purpose
produces complex code using rigorous code style/conventions and full user, technical and in-code documentation	produces complex code using formal style/conventions and documentation	produces code using basic style/conventions and documentation
creates detailed, efficient and well-structured Java code with multiple complex combinations of programming constructs.	creates Java code that includes complex combinations of basic programming constructs.	creates Java code that include basic programming constructs.

Criterion 3: use appropriate objects in the design of programs

This criterion is both internally and externally assessed.

Rating 'A'
In addition to the standards for a 'C' and a 'B' rating, the learner:

Rating 'B'
In addition to the standards for a 'C' rating, the learner:

Rating 'C'
The learner:

Rating A	Rating B	Rating C
writes correctly functioning programs using predefined and self-designed objects	writes correctly functioning programs using a variety of predefined objects	writes correctly functioning programs using a limited number of predefined objects
correctly manipulates values in objects they have defined	correctly manipulates values in complex objects (e.g. arrays)	correctly manipulates values in simple objects (e.g. strings)
defines a new method for a given object, to meet a specification (e.g. to return a result based on numeric, logical or string operations on stored values within that object)	correctly manipulates stored values for an object they declare and instantiate, given its class definition	correctly manipulates stored values for an object, given the method definition for that object
given specifications for an object, creates a class definition for that object.	given definition for a previously unseen object, traces the results of applying the object's methods on the values stored within that object.	traces results of applying an predefined object's methods on the values stored within that object, and determines outputs.

Criterion 4: describe and apply knowledge of computer architecture

This criterion is both internally and externally assessed.

Rating 'A'
In addition to the standards for a 'C' and a 'B' rating, the learner:

Rating 'B'
In addition to the standards for a 'C' rating, the learner:

Rating 'C'
The learner:

Rating A	Rating B	Rating C
interrelates operating systems, machine architecture, fetch, code and execute cycle, machine code, binary, logical expressions and digital circuits	analyses the information flow within a computer, and relates hardware functions to higher order language constructs by converting between machine code and Java	describes basic components of a computer and their function
analyses and provides extensive justification of limitations that computer architecture imposes on computing tasks being performed	describes and gives some justification of limitations that computer architecture imposes on computing tasks being performed	describes limitations that computer architecture imposes on computing tasks being performed
accurately creates complex logic circuits utilising truth tables, Karnaugh maps, logical laws and logic statements expressed in everyday English	accurately creates simple logic circuits utilising truth tables, Karnaugh maps, logical laws and logic statements expressed in everyday English	draws simple logic circuits that correctly represent Boolean expressions and traces them with values
relates TOY machine language operations to their equivalent in Java	completes simple programs in the TOY machine language	traces simple programs in the TOY machine language
analyses relationships between machine language and machine architecture.	describes the significance of the JVM and its relationship to computer architecture.	describes the role of the Java Virtual Machine (JVM).

Criterion 5: analyse how data are represented and stored

This criterion is both internally and externally assessed.

Rating 'A'
In addition to the standards for a 'C' and a 'B' rating, the learner:

Rating 'B'
In addition to the standards for a 'C' rating, the learner:

Rating 'C'
The learner:

Rating A	Rating B	Rating C
evaluate implications of the representation of more complex data types within a computer system and the associated limitations	analyse implications of the representation of all primitive data types within a computer system and the limitations of each	describes the representation of primitive data types in the computer
evaluates characteristics of complex data types and storage issues associated with more complex data structures	explains the special issues associated with floating point representation in the computer and how to deal with them in a program	describes and implements calculation and storage situations where the limitations of the representation of integers has an effect
analyses low-level data storage situations, performs calculations and applies knowledge of data representation to new situations.	performs binary arithmetic, converts between number bases, and can demonstrate relationships between word length and the range of values that can be represented.	accurately performs basic arithmetic in binary and two's complement representation.

Criterion 6: apply the software development life cycle to a variety of problems

Rating 'A'
In addition to the standards for a 'C' and a 'B' rating, the learner:

Rating 'B'
In addition to the standards for a 'C' rating, the learner:

Rating 'C'
The learner:

Rating A	Rating B	Rating C
writes well designed complex programs using the SDLC which meet the specifications using programming standards with an appropriate user interface	follows the Software Development Life Cycle (SDLC) to write complex programs that meet the specifications using the appropriate programming standards	writes straightforward programs which meet specifications using appropriate programming standards and a range of programming constructs
evaluates possible programming constructs and resources, and chooses the most appropriate	uses appropriate programming constructs and accesses relevant resources	accesses and applies core support resources to assist in writing programs
specifies comprehensive testing plans before a program is written, uses both hand and automated tracing to debug programs and refines programs in response to the testing	specifies detailed testing plans before a program is written and makes some program revisions after testing	tests programs against a plan and assesses how well the programs perform
tests and evaluates options for the user interface for a solution, and selects the most appropriate.	explores options for the user interface with regard to the specified problem.	correctly follows a given set of design principles for the user interface.

Criterion 7: analyse and apply societal, professional and ethical responsibilities

Rating 'A'
In addition to the standards for a 'C' and a 'B' rating, the learner:

Rating 'B'
In addition to the standards for a 'C' rating, the learner:

Rating 'C'
The learner:

Rating A	Rating B	Rating C
analyses roles and responsibilities of a range of occupations or careers (including those outside of core computing careers) and evaluates ways in which computer science could contribute to them now and in the future	compares and analyses a variety of occupations involving computer science, including roles and the responsibilities of people working within these occupations	describes a variety of occupations involving computer science, including the roles and the responsibilities of people working within these occupations
applies a professional code of conduct to their own practice (e.g. considers the impact of their own programs on privacy, respects intellectual property, considers the role of their work in enhancing the lives of others)	given a particular scenario, identifies the potential for misuse and relates this potential to professional organisations and codes of ethics	identifies relevant professional organisations and interprets codes of ethics
given a particular scenario, identifies and justifies strategies to be put in place to reduce potential risks from security breaches, user errors and programming errors	applies knowledge of program design and testing to minimise risk of potential technological errors	gives examples of technological errors and the consequences of the errors across a range of areas
selects and applies appropriate standards and conventions when developing own software products, and incorporates user testing with a diverse user group to further inform design decisions.	analyses good and bad user interface design and the impact on users from a diverse range of abilities and backgrounds.	describes purpose and nature of standards and the need to comply with standards (for example, in relation to interface design and accessibility).

Criterion 8: apply personal skills to plan, organise and complete activities

The learner:

Rating A	Rating B	Rating C
proposes and negotiates measurable, achievable and realistic complex goals	proposes and negotiates measurable, achievable and realistic goals	proposes and negotiates achievable and realistic goals
identifies time, resources and equipment needed to complex projects, and develops a formal, systematic and coherent project plan	identifies time, resources and equipment needed to complete complex projects, and develops and employs a formal, coherent project plan	identifies time, resources and equipment needed to complete simple projects, and develops and employs a project plan
meets specified/negotiated timelines and thoroughly addresses all project or task requirements with a high degree of accuracy	meets specified/negotiated timelines and addresses all project or task requirements	meets specified/negotiated timelines and addresses key project or task requirements
reflects - orally and in writing - on progress towards meeting goals and timelines; critically evaluates progress and plans effective future actions.	reflects - orally and in writing - on progress towards meeting goals and timelines; analyses progress to plan future actions.	reflects - orally and in writing - on progress towards meeting goals and timelines, articulating some ways in which goals may be met in the future.

Criterion 9: communicate technological information

In relation to the study of computer science, the learner:

Rating A	Rating B	Rating C
selects, constructs and uses appropriate written, oral, multimodal and mathematical representations to accurately and effectively convey meaning, adapting representations to specific audiences and purposes	selects, constructs and uses appropriate written, oral, multimodal and mathematical representations to produce clear responses for given audience	uses and constructs written, oral, multimodal and mathematical representations as directed that address the basic intent of a question or issue
communicates complex ideas and explanations coherently, selecting and consistently using appropriate language conventions for specific audiences and purposes (within technical, user and in-code documentation)	communicates ideas and explanations clearly, selecting and consistently using appropriate language conventions (within technical, user and in-code documentation)	communicates basic ideas and explanations clearly, correctly using appropriate language conventions (within technical, user and in-code documentation)
clearly differentiates the information, images, ideas and words of others from the learner's own	clearly differentiates the information, images, ideas and words of others from the learner's own	differentiates the information, images, ideas and words of others from the learner's own
referencing conventions and methodologies are followed with a high degree of accuracy	referencing conventions and methodologies are followed correctly	referencing conventions and methodologies are generally followed correctly
creates appropriate, well-structured reference lists/ bibliographies.	creates appropriate, structured reference lists/bibliographies.	creates appropriate reference lists/bibliographies.

Qualifications Available

Computer Science Level 3 (with the award of):

This is the range of awards that learner can achieve. The full range is five awards:

EXCEPTIONAL ACHIEVEMENT
HIGH ACHIEVEMENT
COMMENDABLE ACHIEVEMENT
SATISFACTORY ACHIEVEMENT
PRELIMINARY ACHIEVEMENT

Award Requirements

The final award will be determined by the Office of Tasmanian Assessment, Standards and Certification from 14 ratings (9 from the internal assessment, 5 from external assessment).

The minimum requirements for an award in Computer Science Level 3 are as follows:

EXCEPTIONAL ACHIEVEMENT (EA)
12 'A', 2 'B' ratings (4 'A', 1 'B' from external assessment)

HIGH ACHIEVEMENT (HA)
6 'A', 6 'B', 2 'C' ratings (2 'A', 2 'B', 1 'C' from external assessment)

COMMENDABLE ACHIEVEMENT (CA)|
8 'B', 5 'C' ratings (2 'B', 2'C' ratings from external assessment)

SATISFACTORY ACHIEVEMENT (SA)
12 'C' ratings (3 'C' from external assessment)

PRELIMINARY ACHIEVEMENT (PA)
6 'C' ratings

A learner who otherwise achieves the ratings for a CA (Commendable Achievement) or SA (Satisfactory Achievement) award but who fails to show any evidence of achievement in one or more criteria ('z' notation) will be issued with a PA (Preliminary Achievement) award.

Course Evaluation

The Department of Education's Curriculum Services will develop and regularly revise the curriculum. This evaluation will be informed by the experience of the course's implementation, delivery and assessment.In addition, stakeholders may request Curriculum Services to review a particular aspect of an accredited course.

Requests for amendments to an accredited course will be forwarded by Curriculum Services to the Office of TASC for formal consideration.

Such requests for amendment will be considered in terms of the likely improvements to the outcomes for learners, possible consequences for delivery and assessment of the course, and alignment with Australian Curriculum materials.

A course is formally analysed prior to the expiry of its accreditation as part of the process to develop specifications to guide the development of any replacement course.

Course Developer

The Department of Education acknowledges the significant leadership of Mr Bruce Stack, Mr Rob Torok, Mr Grant MacDonald and Dr Ken Price in the development of this course.

Expectations Defined By National Standards

There are no statements of national standards relevant to this course.

Accreditation

The accreditation period for this course has been renewed from 1 January 2019 until 31 December 2021.

During the accreditation period required amendments can be considered via established processes.

Should outcomes of the Years 9-12 Review process find this course unsuitable for inclusion in the Tasmanian senior secondary curriculum, its accreditation may be cancelled. Any such cancellation would not occur during an academic year.

Version History

Version 1 – Accredited on 30 July 2017 for use from 1 January 2018. This course replaces Computer Science (ITC315113) that expired on 31/12/2017.

Accreditation renewed on 22 November 2018 for the period 1 January 2019 until 31 December 2021.

Version 2 - Accreditation renewed on 14 July 2021 for the period 1 January 2022 until 31 December 2023. Removal of reference to Java applets, Java applications now compulsory. See Learning Outcomes, Content (Programming and Area 4 Option), and 'A' rating descriptor for Criterion 1, Element 3).

Appendix

REFERENCES

ACS/DAE, Australia's Digital Pulse Developing the digital workforce to drive growth in the future 2016, Australian Computer Society/Deloitte Access Economics, Sydney, NSW.
Viewed Oct 2016, http://www2.deloitte.com/content/dam/Deloitte/au/Documents/Economics/deloitte-au-economics-digital-pulse-2016-acs-110316.pdf

Line Of Sight

Learning Outcome	Criterion/ia	Criteria Elements Content	Content
· apply the techniques of computer science to a range of problems and appreciate the limitations of using algorithmic solutions	1, 2	1E1, 1E2, 1E3, 1E4, 1E5, 2E1, 2E5	Problem Solving and Programming
· describe the interaction between people and computers, and the implications for software design	6, 7	6E4,7E3, 7E4	Social Ethical Issues and Professional Responsibility
· design algorithmic, programming and technological solutions to a range of problems expressed in a variety of forms	1, 2, 3, 6	1E1, 1E2, 1E3, 1E4, 1E5, 2E1, 2E2, 2E3, 2E4, 2E5, 3E1, 3E2, 3E3, 3E4, 6E1, 6E2, 6E3, 6E4	Problem Solving and Programming
· compare, evaluate and refine algorithmic and programming solutions	1, 2, 3, 6	1E1, 1E2, 1E3, 1E4, 1E5, 2E1, 2E2, 2E3, 2E4, 2E5, 3E1, 3E2, 3E3, 3E4, 6E1, 6E2, 6E3, 6E4	Problem Solving and Programming
· describe software and hardware aspects of computing, and explain their operation with the underpinning mathematics and science of the discipline	4, 5	4E1, 4E2, 4E3, 4E4, 4E5, 5E1, 5E2, 5E3	Computer Fundamentals and Computer Limitations
· use a variety of technological resources such as online libraries and technical websites	7, 9	7E1, 7E2, 7E3, 7E4, 9E3, 9E4	Social Ethical Issues and Professional Responsibility
· describe societal consequences of technological solutions and the professional responsibility of people working within this field	7	7E1, 7E2, 7E3, 7E4	Social Ethical Issues and Professional Responsibility
· provide effective communication to a range of stakeholders about technical problems and their solutions	9	9E1, 9E2, 9E3, 9E4	Problem Solving and Programming
· manage their own learning, including time management and organisation skills	8	8E1, 8E2, 8E3, 8E4	Computing Option
· identify career and further education opportunities that make use of computer science skills, knowledge and understanding	7	7E1, 7E2	Social Ethical Issues and Professional Responsibility

Supporting documents including external assessment material

- ITC315118 Information Booklet 2018.pdf (2018-10-31 11:02am AEDT)
- ITC315118 Computer Science TASC Exam Paper 2018.pdf (2018-12-09 10:02am AEDT)
- ITC315118 - Assessment Panel Report 2018.pdf (2019-02-27 09:28am AEDT)
- ITC315118 Computer Science TASC Exam Paper 2019.pdf (2019-11-18 08:45am AEDT)
- ITC315118 Assessment Report 2019.pdf (2020-01-24 02:56pm AEDT)
- ITC315118 Computer Science TASC Exam Paper 2020.pdf (2020-11-16 10:52pm AEDT)
- ITC315118 Assessment Report 2020.pdf (2021-01-18 10:32am AEDT)
- ITC315118 Computer Science TASC Exam Paper 2021.pdf (2021-11-20 04:36pm AEDT)
- ITC315118 Assessment Report 2021.pdf (2022-01-24 12:56pm AEDT)
- ITC315118 Computer Science External Assessment Specifications.pdf (2022-04-07 03:27pm AEST)
- TASC ITC315118 Computer Science Exemplar Questions 2022.pdf (2022-05-16 10:31am AEST)

Tasmanian Government