SQA Advanced Unit Specification

General information

Unit title: Machines, Languages & Computation (SCQF level 7)

Unit code: HT08 47

Superclass: CB

Publication date: August 2017

Source: Scottish Qualifications Authority

Version: 01

Unit purpose

The purpose of this unit is to introduce learners to the fundamentals of computation and computational thinking. It also introduces foundational mathematical concepts that are central to the study of Computer Science.

The topics covered are: algorithms, functions, mathematical proofs, recursion, propositional logic and an introductory account of abstract machines. Learners are also encouraged to develop their computational thinking skills.

The Knowledge and Skills gained in this unit underpin many other areas of Computer Science, with notable examples being software engineering, software verification, hardware design, programming language design and artificial intelligence.

The unit is suitable for learners, with appropriate backgrounds in computing and mathematics, who wish to better understand foundation concepts in these fields for the purpose of progression to higher level studies in Computer Science or a related discipline.

Outcomes

On successful completion of the unit the learner will be able to:

1 Construct non-complex algorithms.
2 Create recursive rules to generate specified languages, algorithms and data structures.
3 Derive theorems in propositional logic using symbolic rules.
4 Create abstract machines to perform computations.

Credit points and level

2 SQA Credits at SCQF level 7: (16 SCQF credit points at SCQF level 7)

Recommended entry to the unit

Access to this unit will be at the discretion of the centre, however it is recommended that learners should have some prior knowledge and skill in Computing/IT demonstrated by achievement of relevant National courses or units at National 5 or Higher, or appropriate work experience. Because the Core Skill of *Numeracy* is embedded in this unit it is also recommended that learners should have some prior knowledge and skill in mathematics demonstrated by achievement of National 5 or Higher.

Core Skills

Opportunities to develop aspects of Core Skills are highlighted in the support notes for this unit specification.

There is no automatic certification of Core Skills or Core Skill components in this unit.

Context for delivery

If this unit is delivered as part of a group award, it is recommended that it should be taught and assessed within the subject area of the group award to which it contributes.

The Assessment Support Pack (ASP) for this unit provides assessment and marking guidelines that exemplify the national standard for achievement. It is a valid, reliable and practicable assessment. Centres wishing to develop their own assessments should refer to the ASP to ensure a comparable standard. A list of existing ASPs is available to download from SQA's website (http://www.sqa.org.uk/sqa/46233.2769.html).

Equality and inclusion

This unit specification has been designed to ensure that there are no unnecessary barriers to learning or assessment. The individual needs of learners should be taken into account when planning learning experiences, selecting assessment methods or considering alternative evidence.

Further advice can be found on our website www.sqa.org.uk/assessmentarrangements.

Unit Specification: Statement of standards

Unit title: Machines, Languages & Computation (SCQF level 7)

Acceptable performance in this unit will be the satisfactory achievement of the standards set out in this part of the unit specification. All sections of the statement of standards are mandatory and cannot be altered without reference to SQA.

Where evidence for outcomes is assessed on a sample basis, the whole of the content listed in the Knowledge and/or Skills section must be taught and available for assessment. Learners should not know in advance the items on which they will be assessed and different items should be sampled on each assessment occasion.

Outcome 1

Construct non-complex algorithms.

Knowledge and/or Skills

- Definition of algorithm
- Methods of expressing algorithms
- Set theory
- Boolean algebra
- Mathematical functions
- Functions and non-functions
- Data types including Booleans and lists
- Algorithms for performing common tasks
- Computational thinking

Outcome 2

Create recursive rules to generate specified languages, algorithms and data structures.

Knowledge and/or Skills

- Recursion
- Recursive rules
- Context free grammars
- Recursive data structures including lists and trees
- Recursive algorithms for processing lists and integers

Outcome 3

Derive theorems in propositional logic using symbolic rules.

Knowledge and/or Skills

- Propositional logic
- Symbolic rules
- Formal proofs
- Soundness and completeness in a symbolic system
- Conjunctive normal form

Outcome 4

Create abstract machines to perform computations.

Knowledge and/or Skills

- Finite state machines
- Formal languages
- Turing Machine
- Moore and Mealy Machines

Evidence requirements for this unit

Candidates will need to provide evidence to demonstrate their Knowledge and/or Skills across all outcomes.

Evidence is normally required for all of the Knowledge and Skills. However, sampling may be used in a specific circumstance (see below).

The evidence requirements for this unit will take the form of evidence of **cognitive competence** (for all outcomes). It may be written or oral or a combination of these. Evidence may be captured, stored and presented in a range of media (including audio and video) and formats (analogue and digital).

The amount of evidence should be the minimum consistent with the stated Knowledge and Skills. It is up to the assessor to decide when a competence has been adequately demonstrated. In some cases, such as the definition of algorithms, a single demonstration is adequate; in most cases, multiple demonstrations will be necessary.

The evidence (for all outcomes) may be wholly or partly produced under controlled conditions. When evidence is produced in uncontrolled or loosely controlled conditions it must be authenticated. The *Guide to Assessment* provides further advice on methods of authentication.

There are no time limitations on the production of evidence (but see exception below). The evidence may be produced at any time during the life of the unit. Learners may use reference materials when undertaking assessment (but see exception below).

Sampling is permissible when the evidence is produced through testing. The test may take any form (including oral) but must be supervised, unseen and timed. The contents of the test must sample broadly and proportionately from the contents of Outcomes 1, 2, 3 and 4 with approximately equal weighting for each outcome. Access to reference material is not appropriate for this type of assessment.

Evidence must be consistent with the SCQF Level of this unit. The following SCQF level descriptors are particularly relevant to this unit and may be applied to the evidence when appropriate:

- Knowledge is embedded in theories, concepts and principles
- Apply knowledge and understanding in routine and non-routine contexts
- Present and evaluate arguments, information and idea that are routine
- Use a range of approaches to address defined/routine problems within familiar contexts
- Convey complex ideas in well-structured and coherent form
- Use a range of numerical and graphical techniques skills in combination

The *Guidelines on Approaches to Assessment* (see the support notes section of this specification) provides specific examples of instruments of assessment.

Unit Specification: Support notes

Unit title: Machines, Languages and Computation (SCQF level 7)

Unit support notes are offered as guidance and are not mandatory.

While the exact time allocated to this unit is at the discretion of the centre, the notional design length is 80 hours.

Outcome 1: 20 hours
Outcome 2: 20 hours
Outcome 3: 20 hours
Outcome 4: 20 hours

Guidance on the content and context for this unit

This unit is based on a selection of material from the class CS103: Machines, Languages and Computation delivered at the University of Strathclyde. It was designed to be used as part of a set of units for an SQA Advanced Certificate which prepare the learner for possible 2nd year direct entry to university. It can also be used as foundational material for all courses in Computer Science, Computing and Software Engineering. Users of this material should note that CS103 contains more material than this unit and therefore should not use any CS103 assessment materials as being indicative of the requirements of this unit.

Outcome 1

This outcome introduces the learner to the fundamental concepts of algorithms and mathematical functions. The learning material can be delivered in the form of lectures and tutorial exercises, and could be usefully supplemented with short introductory programming exercises. A good language to use for this purpose would be Python.

Outcome 2

This outcome introduces the fundamental Computer Science technique of recursion and explores its use in defining grammars, data structures and algorithms. The material can be delivered in the form of lectures and tutorial exercises, and could be usefully supplemented with short introductory programming exercises involving recursion.

Outcome 3

This outcome introduces propositional logic as a symbolic system and shows how to prove theorems using symbolic rules. The fundamental ideas of satisfiability, soundness and completeness are introduced. The learner is also shown how to convert a given formula into conjunctive normal form. The material can be delivered in the form of lectures and tutorial exercises.

Outcome 4

This outcome is concerned with abstract machines as models of computation, and the idea that different types of automata, ranging from finite state machines to Turing machines, are needed to model increasingly complex forms of computation. The material can be delivered in the form of lectures and tutorial exercises.

Guidance on approaches to delivery of this unit

See above. It is recommended that the outcomes be taught in the order given. The material is mathematical in nature, and so the approach should be based on presentation of techniques followed by engagement with practical problem solving exercises.

Guidance on approaches to assessment of this unit

Evidence can be generated using different types of instruments of assessment. The following are suggestions only. There may be other methods that would be more suitable to candidates.

A traditional approach to evidence generation for this unit would be to assess candidates by a test of their knowledge and understanding. The test could be of two hours duration and could consist of short questions designed to test each of the defined Knowledge and Skills. Sampling would be permitted (for example, for the first item of Outcome 1, an algorithm may be presented using a flowchart, pseudocode or unambiguous natural language, but it is not a requirement to explicitly cover all three forms of presentation in the test). The pass mark for the exam could be set at 40%.

Outcome 1

Evidence for the Knowledge and/or Skills in this outcome will be provided by an assessed piece of coursework covering the following topics:

- Explain what an algorithm is and why it is of such importance in Computer Science.
- Given an example of a simple algorithm expressed as a flowchart, pseudocode or unambiguous natural language, say what the output from the algorithm would be for a given input.
- Create algorithms for various simple functions using pseudocode, flow charts or unambiguous English. The functions can include lists and integers are inputs and lists, integers and Booleans as outputs.
- Given an association between two sets, say whether or not that association is a function.
- Given a simple mathematical function relating sets of integers, write an algorithm for that function.

Outcome 2

Evidence for the Knowledge and/or Skills in this outcome will be provided by an assessed piece of coursework covering the following topics:

- Generate sentences using a grammar. Modify a given grammar so that it can generate specified sentences.
- Generate strings using a grammar. Write a grammar which can generate strings given regular expressions.
- Recognise recursion in simple functional programs and recognise the difference between iterative code and recursive code.
- Given a recursive function and an argument, draw a diagram showing how the function is called and returns its results.
- Write a simple recursive algorithm to implement a given function.

Outcome 3

Evidence for the Knowledge and/or Skills in this outcome will be provided by an assessed piece of coursework covering the following topics:

- Construct a truth table for a propositional formula.
- State whether a propositional formula is a tautology or not and whether it is a contradiction.
- For a given propositional formula, give a proof of the formula using natural deduction.
- Rewrite a propositional formula in CNF (conjunctive normal form) and say whether the formula is satisfiable.

Outcome 4

Evidence for the Knowledge and/or Skills in this outcome will be provided by an assessed piece of coursework covering the following topics:

- Describe the language recognised by a given deterministic finite state machine.
- Construct a deterministic finite state machine for a given specification.
- Describe the specification for a given Moore or Mealy machine.
- Construct a Moore or Mealy machine for a given specification.
- Describe the behaviour of a given Turing machine.
- Construct a Turing machine for a given specification.

Centres are reminded that prior verification of centre devised assessments would help to ensure that the national standard is being met. Where learners experience a range of assessment methods, this helps them to develop different skills that should be transferable to work or further and higher education.

Opportunities for e-assessment

E-assessment may be appropriate for some assessments in this unit. By e-assessment we mean assessment which is supported by Information and Communication Technology (ICT), such as e-testing or the use of e-portfolios or social software. Centres which wish to use e-assessment must ensure that the national standard is applied to all learner evidence and that conditions of assessment as specified in the evidence requirements are met, regardless of the mode of gathering evidence. The most up-to-date guidance on the use of e-assessment to support SQA's qualifications is available at www.sqa.org.uk/e-assessment.

Opportunities for developing Core and other essential skills

This unit should aid in the development of mathematical and computational thinking skills.

History of changes to unit

Version	Description of change	Date

SQA acknowledges the valuable contribution that Scotland's colleges have made to the development of SQA Advanced Qualifications.

FURTHER INFORMATION: Call SQA's Customer Contact Centre on 44 (0) 141 500 5030 or 0345 279 1000. Alternatively, complete our Centre Feedback Form.

General information for learners

Unit title: Machines, Languages & Computation (SCQF level 7)

This section will help you decide whether this is the unit for you by explaining what the unit is about, what you should know or be able to do before you start, what you will need to do during the unit and opportunities for further learning and employment.

The purpose of this unit is to introduce some of the fundamental concepts and mathematical techniques used in Computer Science: algorithms, functions, recursion, formal languages, data structures, propositional logic, symbolic systems and abstract machines as models of computation.

The unit will mostly be delivered through lectures and pen-and-paper exercises, although some programming may be involved. No previous knowledge of programming is needed.

Because of the mathematical content, you should be qualified in mathematics at least up to National 5 or an equivalent standard.

You will be assessed through four pieces of coursework that are submitted throughout the unit, with a single final exam covering all aspects of the unit at the end.

On completion of the unit, you will be competent to study theoretical Computer Science at the standard required for 2nd year of a Scottish University course. You will also have gained foundational skills in computational thinking and mathematics.