Next Generation Higher National Unit Specification

Applied Mathematics for Computing (SCQF level 8)

Unit code: J7DT 48

SCQF level: 8 (16 SCQF credit points)

Valid from: session 2023–24

Prototype unit specification for use in pilot delivery only (version 1.0) June 2023

This unit specification provides detailed information about the unit to ensure consistent and transparent assessment year on year.

This unit specification is for teachers and lecturers and contains all the mandatory information required to deliver and assess the unit.

This edition: June 2023 (version 1.0)

Unit purpose

This unit introduces learners to the key mathematical concepts that relate to algorithms and computation. Learners use software and programming approaches to implement mathematical routines to solve problems.

This specialist unit is designed for learners with an interest in mathematical foundations relevant to the study of computer science. It is particularly suitable for learners with a vocational interest in STEM-based subjects, or who want to progress to further levels of qualification. Before starting the unit, learners should have knowledge of discrete mathematics topics at SCQF level 7 and a basic understanding of computer programming.

On completion of the unit, learners:

- understand the formal reasoning for sets, relations, functions and cardinality
- can represent and reason problems using graphs
- use matrices and transformations to define and manipulate graphical objects
- have skills in the use of probability and statistics in analysis
- may progress to other mathematical topics at SCQF level 8, such as the applications of discrete mathematics concepts in STEM-related subjects

Unit outcomes

Learners who complete this unit can:

1 explain sets, logic and their application in computing
2 use counting methods, including permutations and combinations
3 explain the solution methods of matrices and graphs as they arise in computer science problems
4 explain the solution methods of recurrence relations as they arise in computer science problems
5 solve problems in computing using probability and statistics

Evidence requirements

The unit requires both knowledge and product evidence. Learners' knowledge evidence must include the solution to at least one problem in each of these:

- both permutation and combination
- propositional logic
- truth table and bitwise operations
- set operations, including set-builder notation
- prime numbers
- greatest common denominator
- number theory applications in cryptography
- growth function and Big O notation
- matrix operations including linear mapping
- sequences and recursive definitions
- mathematical induction (sequences)
- graph representation and graph properties
- probability distributions
- statistical estimation and hypothesis testing

Learners' knowledge evidence must be produced under controlled conditions in terms of supervision, location, timing and access to reference materials.

Learners' product evidence must include:

- representing an algorithm in program code
- performing a logic operations in program code
- plotting a growth functions using software
- performing an operation on matrices in program code
- creating a recursive algorithms in program code
- performing a graph operations using software
- plotting a probability density function using software

- performing a statistical estimation using software
- performing least-squares regression on given data

Learners can produce product evidence over an extended period in lightly controlled conditions. Evidence produced in lightly controlled conditions must be authenticated. The Guide to Assessment provides further advice on methods of authentication.

Knowledge and skills

The following table shows the knowledge and skills covered by the unit outcomes:

Knowledge	Skills
Learners should understand: ♦ counting: — the pigeonhole principle — permutations and combinations — binomial coefficients ♦ logic and algorithms: — propositional logic — truth tables — predicates and qualifiers ♦ applications of logic in computing, including Bitwise operations and circuits ♦ set theory: — set operations — sets as lists — set-builder notation — countability — orderings ♦ number theory: — divisibility — integer representations — primes and greatest common denominators — cryptography applications ♦ functions: — growth functions — Big O notation	Learners can: ♦ represent an algorithm in program code ♦ represent a truth table in program code ♦ perform operations on sets using program code ♦ perform a set of logic operations in code ♦ visualise functions by plotting, using software ♦ perform matrix operations using software ♦ create recursive algorithms using program code ♦ perform graph operations using software ♦ plot probability density functions using software ♦ perform statistical estimation using software ♦ perform least-squares regression on given data

Knowledge	Skills
Learners should understand: ♦ matrices and linear equations: — matrix properties — matrix addition and multiplication — inverse matrices and determinants — linear mappings in r^2 (rotation and reflection) ♦ sequences and series: — sequences and sigma notation — recursive definitions and algorithms — mathematical induction ♦ graph theory: — representation and adjacency matrix — weighted graphs — walks, trails, paths and cycles — connectivity — trees ♦ probability and descriptive statistics: — probability distributions — discrete and continuous random variables — statistical estimation and least squares — applications to errors, noise, and least squares fitting of straight lines and other curves to data — hypothesis testing	

Meta-skills

Throughout this unit, learners develop meta-skills to enhance their employability in the computer science sector.

Self-management

This meta-skill includes:

- focusing: concentrating on abstract mathematical concepts
- adapting: applying concepts to real-world problems in computer science that demand logic and resilience

Social intelligence

This meta-skill includes:

- communicating: receiving complex information, applying it, and communicating results to others
- feeling: demonstrating empathy when addressing conceptual misunderstandings
- collaborating: working collaboratively with others to find solutions to problems and sharing results

Innovation

This meta-skill includes:

- curiosity: investigating how mathematical concepts relate to practical computing problems
- critical thinking: acquiring new ways of addressing problems and visualising how mathematical structures arise in computing

Delivery of unit

The time required to deliver this unit varies depending on the previous experience of individual learners.

Based on 80 hours of delivery and assessment time, we suggest the following distribution:

Outcome 1 — Demonstrate an understanding of sets, logic and their application in computing
(10 hrs)

Outcome *2* — Use counting methods including permutations and combinations
(10 hrs)

Outcome 3 — Understand the solution methods of matrices and graphs as they arise in computer science problems
(20 hrs)

Outcome 4 — Understand the solution methods of recurrence relations as they arise in computer science problems
(10 hrs)

Outcome 5 — Use knowledge of probability and statistics to solve problems in computing
(30 hrs)

Additional guidance

The guidance in this section is not mandatory.

Content and context for this unit

This unit prepares learners to progress to computer science qualifications that require more advanced mathematical knowledge and skills. Learners apply logical and computational thinking to use mathematical concepts and constructs to solve problems. The skills of analysis and synthesis are key components of learning in the unit.

You should revisit some basic mathematical concepts to ensure a sound foundation before introducing new concepts.

You should emphasise the connection between the mathematical concept or construct and its representation as a real-world computer science problem, such as counting or logical reasoning. Wherever possible, learners should consider how they can represent the application of a mathematical technique or solution in program code, using a suitable programming language.

You should only go as far as covering r-combinations with repetition when talking about permutations and combinations. You do not need to teach the multinomial case.

You should only cover binary relations when teaching about sets. Ask learners to consider sequences (arithmetic, geometric) and recurrence relations. The solution of recurrence relations should include iterative solutions. You should cover the application of recurrence relations to binary search, as well as the use of induction to find the solution.

Graphs

You should begin with undirected graphs before moving to directed, and then illustrate the range of graphs found in computer science problems. The representation of graphs in adjacency matrices should lead to the concepts of paths, cycles and trails and their practical applications before you introduce trees.

Statistics

You should begin with basic probability theory and types of variables before introducing probability distributions. You should cover the treatment of continuous and discrete data, along with statistical summary measures, leading to basic hypothesis testing. You should teach correlation and regression in a practical way.

Approaches to assessment

Knowledge evidence should be obtained by a question paper that includes all of the problem types set out in the evidence requirements above. The pass mark should be set at 70%. The question paper could use one or more questions for each problem type, while ensuring coverage of the concepts introduced in the unit.

You can decide what type of tasks are to be used to generate the product evidence. The program code produced for tasks should be documented to a required standard and include a test plan along with test results. The software to be used to generate statistical summaries and estimates should be that used in the delivery of the content, or an alternative of similar functionality. A short report should be produced that evaluates the use of software approaches to solve applied mathematics problems that arise in the context of computer science.

Equality and inclusion

This unit is designed to be as fair and as accessible as possible with no unnecessary barriers to learning or assessment.

You should take into account the needs of individual learners when planning learning experiences, selecting assessment methods or considering alternative evidence.

Guidance on assessment arrangements for disabled learners and/or those with additional support needs is available on the assessment arrangements web page: www.sqa.org.uk/assessmentarrangements.

Information for learners

Applied Mathematics for Computing (SCQF level 8)

This information explains:

- what the unit is about
- what you should know or be able to do before you start
- what you need to do during the unit
- opportunities for further learning and employment

Unit information

This specialist unit is designed for learners with an interest in mathematical foundations relevant to the study of computer science. It is particularly suitable for you if you have a vocational interest in STEM-based subjects, or you wish to progress to higher education. Before starting the unit, you should have previous knowledge of discrete mathematics topics at SCQF level 7, along with a basic understanding of computer programming.

In this unit, you:

- are introduced to the key mathematical concepts that relate to algorithms and computation
- learn how to use software and programming approaches to implement mathematical routines to solve problems
- develop an understanding of the formal reasoning for sets, relations, functions and cardinality
- consider the representation and reasoning of problems using graphs and the use of matrices and transformations for defining and manipulating graphical objects
- examine the usage of probability and statistics in analysis

You are assessed through a test of mathematical knowledge and understanding, most likely in the form of a test and practical evidence in which you can apply statistical methods to find the solution of a problem.

Throughout the unit, you develop meta-skills covering self-management, social intelligence and innovation.

On completion of the unit, you may progress to other mathematical topics at SCQF level 8 and higher.

Administrative information

Published: June 2023 (version 1.0)

Superclass: RB

History of changes

Version	Description of change	Date

Note: please check SQA's website to ensure you are using the most up-to-date version of this document.