Next Generation Higher National Unit Specification

Computer Programming: Applied Mathematics (SCQF level 7)

Unit code: J7S2 47

SCQF level: 7 (8 SCQF credit points)

Valid from: session 2023–24

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

Unit purpose

This unit enables learners to create program code, using a high-level language to solve numerical, statistical and mathematical problems. It provides learners with underpinning knowledge of numbers, statistics and mathematics, and develops their coding skills. Learners develop and apply this knowledge in the context of coding functions and algorithms. They can use any suitable high-level programming language. The unit develops the following underpinning concepts in applied mathematics for computing:

- number types (including complex numbers)
- number systems (including hexadecimal)
- scientific notation
- factorials
- measures of dispersion (including mean absolute deviation)
- probability
- Boolean algebra and set theory
- trigonometry

Learners demonstrate the processes for manipulating and solving problems relating to these concepts in a programming context. This helps them to develop a library of numerical, statistical and mathematical routines using a contemporary high-level language.

This is a non-specialist unit that is in a range of computing awards. It is particularly suitable for learners who require to apply numerical methods in programming contexts, such as learners who want to develop computer games or systems software.

On completion of this unit, learners may progress to further studies in programming or mathematics or progress to a similar unit at a higher level, such as Computer Programming: Applied Mathematics at SCQF level 8.

Unit outcomes

Learners who complete this unit can:

1 write numerical functions in a high-level language
2 write statistical functions in a high-level language
3 write mathematical functions in a high-level language

Evidence requirements

Learners must provide knowledge and product evidence.

The knowledge evidence must cover the required knowledge in each of the outcomes as set out in the knowledge and skills table. Learners must demonstrate the required cognitive competence in number, statistics and mathematics.

You can sample knowledge and understanding when testing is used. The test must sample broadly and widely from the defined knowledge and must be unknown and unpredictable to the learner. When testing is used, learners must produce evidence under controlled conditions in terms of location, supervision and timing. They are not allowed access to reference material if testing is used.

Product evidence for all outcomes relates to the learner's practical competence to design, code and test program functions that perform numerical, statistical and mathematical computations.

This evidence can take any appropriate form, as long as it demonstrates the learner's competence in creating program functions in a high-level language. The program functions cover a range of numerical, statistical and mathematical functions. Learners select the functions that they demonstrate in code. The functions must be non-trivial and sample broadly and widely across all outcomes.

Learners can demonstrate their competence in separate program functions or one large program that incorporates a selection of functions. Evidence may be wholly or partly produced under controlled conditions. When evidence is produced in uncontrolled or lightly-controlled conditions it must be authenticated. The Guide to Assessment provides further advice on methods of authentication.

The amount of evidence should be the minimum consistent with assuring the assessor that the learner possesses each specific knowledge and skill.

Knowledge and skills

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

Knowledge	Skills
Learners should understand: ♦ number types (natural, whole, integer, rational, irrational, real, complex) ♦ fundamental operations ♦ operators and operator priority ♦ percentages, fractions, and decimals ♦ numbering systems (bases) ♦ scientific notation ♦ floating point representation and number precision ♦ powers, roots and factorials ♦ measures of central tendency (including mean) ♦ measures of dispersion (including mean absolute deviation) ♦ probability (including combinations and permutations) ♦ standard error ♦ elementary algebra ♦ Boolean algebra (including Boolean logic operators) ♦ applications of set theory (including set operations) ♦ working with two-dimensional geometry (shapes and points) ♦ linear algebra (two-dimensional vectors) ♦ trigonometric functions (including sine, cosine and arctangent)	Learners can: ♦ use built-in numerical functions and library routines ♦ create numerical expressions and functions in a high-level language ♦ use built-in statistical functions and library routines ♦ create statistical expressions and functions in a high-level language ♦ use built-in mathematical functions and library routines ♦ create mathematical expressions and functions in a high-level language

Meta-skills

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

Self-management

This meta-skill includes:

- focusing: sorting, filtering
- integrity: ethics
- initiative: decision making

Social intelligence

This meta-skill includes:

- communicating: receiving information, giving information
- leading: influencing

Innovation

This meta-skill includes:

- curiosity: information sourcing, problem recognition
- creativity: visualising
- sense-making: pattern recognition, synthesis, analysis
- critical thinking: deconstruction, logical thinking, computational thinking

Delivery of unit

We suggest the following distribution of time:

Outcome 1 — Write numerical functions in a high-level language.
(8 hours)
Outcome 2 — Write statistical functions in a high-level language.
(14 hours)
Outcome 3 — Write mathematical functions in a high-level language.
(18 hours)

Our suggested distribution of time for delivery of this unit is based on the assumption that outcome 1 is fundamental and starts from first principles, outcome 2 involves more practical applications, while outcome 3 contains the largest number of topics.

Additional guidance

The guidance in this section is not mandatory.

Content and context for this unit

We recommend that you deliver the unit content in the order of the outcomes. This recognises that certain aspects of mathematics are interconnected, and learning in certain areas can be useful or even necessary before attempting later techniques. For example, understanding percentages is likely to be necessary when calculating probabilities and odds. However, that does not necessarily mean that you assess the content in that same order. For example, you can assess the basic numerical knowledge later alongside the more technical requirements which they underpin.

We recommend that you deliver the unit using varied approaches to learning and teaching, where possible. However, due to the nature of the content and the wide range of material covered, opportunities for group activities may be limited and might only be feasible for formative work. You should place emphasis on practical tasks, particularly coding, where possible. Although certain elements may require working with mathematical problems in a theoretical setting, you should keep this to a minimum, as paper-based mathematical problems would not be well suited to the learning in this unit.

Approaches to assessment

You can generate evidence using various types of assessment. The following are suggestions only. There may be other methods or approaches that would be more suitable to learners.

You could assess this unit by two assessments: one relating to knowledge and understanding (for evidence of cognitive competence), and one relating to practical abilities (for evidence of practical competence). You can assess knowledge and understanding by using a traditional test that assesses the learner's knowledge and understanding of the defined knowledge domain (across all outcomes). This would be a timed, closed-book test of their knowledge of numerical, statistical and mathematical methods, with an appropriate pass mark (normally 50%).

You can assess practical abilities using a portfolio of programs and/or code segments that demonstrate the learner's ability to write a range of numerical, statistical and mathematical functions. The programs (or code segments) could demonstrate that the learner can write code that:

- involves operators and operator priority, percentages and fractions
- converts between number systems
- works out factorials
- computes mean, media and mode
- evaluates probability
- works out standard deviation
- performs algebraic manipulations
- demonstrates Boolean algebra

- calculates trigonometric functions

Learners decide on the selection of code segments to include in their portfolios. They generate their portfolio throughout the unit, under lightly-controlled conditions. You must use a valid form of authentication (such as oral questioning) to verify that it is the learner's own work.

Where learners experience a range of assessment methods, this helps them to develop a range of skills that should be transferable to work or further study.

An example of one possible scheme for assessment of this unit is as follows, contextualised for delivery as part of a computer games qualification:

Knowledge evidence

You might decide to assess certain topics using a set of 30 multiple-choice questions, or two sets of 15 multiple-choice questions, covering criteria from multiple outcomes. For example:

Assessment 1

- Number types: five questions regarding numeric data types, storage of real and rational numbers, accuracy of real and irrational numbers, effects of mixing data types in an expression, use of complex numbers.
- Numbering systems: five questions regarding decimal, binary and hex conversion.
- Scientific notation: five questions regarding mantissa and exponent, standard form, and orders of magnitude.

Assessment 2

- Probability: five questions on combinations and permutations.
- Boolean algebra: five questions regarding Boolean logic operators, truth tables, and bitwise logic.
- Set theory: five questions regarding set union, intersection, difference, and set data structures in programming languages.

Practical evidence

This can be a portfolio of projects that require learners to program in a high-level language to solve a variety of mathematical and statistical problems. You can provide code outlines for learners to apply mathematical techniques to achieve stated goals. The use of mathematical functions in these solutions fulfils the requirement to use built-in mathematical functions and library routines.

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

Computer Programming: Applied Mathematics (SCQF level 7)

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 is an introductory unit to using mathematical techniques and algorithms in programming. Before starting the unit, you should have a general understanding of computer programming, along with a general level of understanding of mathematical concepts and their applications. The unit introduces you to concepts in numbers, statistics and mathematics that are used in practical programming applications. You can take this unit as part of a computing course or you can study this on its own if you want to improve your programming and mathematics skills.

You are assessed on your knowledge and understanding of the mathematical concepts and methods in the unit, and on your competence in carrying out practical work in coding functions and algorithms relating to these. You are presented with a variety of programming problems to solve using the mathematical and statistical functions available in the high-level programming language you are using. These solutions form a complete or partial portfolio. Your tutor will inform you whether the portfolio should be complete, covering all the unit requirements, or whether it should be partial, with the remaining requirements being met by a written assessment or test.

The programming solutions cover areas such as:

- fundamental numeracy, from basic operations like adding and subtracting, to factorials and roots; as well as the basics of data representation in computers
- statistical functions, including techniques related to finding averages, distributions of data, and probability
- general mathematical functions, in areas such as Boolean logic, set theory, geometry, vector mathematics, and trigonometry

While solving these problems, you develop elements of meta-skills such as self-management, social intelligence and innovation.

On completion of this unit, you may progress to further study in applied mathematics for computing, including Applied Mathematics for Computing at SCQF level 8.

Administrative information

Published: September 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.