National Unit Specification

General information

Unit title: Computer Programming (SCQF level 4)

Unit code: HY2C 44

Superclass: CB

Publication date: March 2018

Source: Scottish Qualifications Authority

Version: 01

Unit purpose

The purpose of this unit is to provide foundational programming skills and a basic knowledge of the principles of computer programming.

This is a **non-specialist** unit, suitable for a **wide range of learners**. It is suitable for learners who require an introduction to coding for vocational purposes, and for learners who wish to appreciate programming for academic or personal reasons. No previous programming experience is required.

Learners will gain a range of practical skills and acquire relevant underpinning knowledge. They will learn how to write code in a contemporary high-level language and appreciate basic programming concepts and techniques, and develop their computational thinking skills.

On completion of this unit, learners will know how to write simple programs to solve real-world problems. Learners will be able to apply foundational programming concepts by implementing them in a programming environment.

Learners may progress to H2YC 45 *Computer Programming* at SCQF level 5.

Outcomes

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

1 Write an algorithm to represent the solution to a simple problem.
2 Describe basic programming concepts.
3 Write a simple computer program.

National Unit Specification: General information (cont)

Unit title: Computer Programming (SCQF level 4)

Credit points and level

1 National Unit credit(s) at SCQF level 4: (6 SCQF credit points at SCQF level 4).

Recommended entry to the unit

Entry is at the discretion of the centre. No previous knowledge or experience of programming is required.

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.

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**.

National Unit Specification: Statement of standards

Unit title: Computer Programming (SCQF level 4)

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 performance criteria 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

Write an algorithm to represent the solution to a simple problem.

Performance criteria

(a) Describe a common method of writing algorithms
(b) Break down a problem into a series of smaller steps
(c) Create an algorithm to solve a simple problem

Outcome 2

Describe basic programming concepts.

Performance criteria

(a) Describe the concepts of syntax and semantics
(b) Describe the concepts of data and instructions
(c) Describe the correct uses of variables, naming conventions and data types
(d) Describe the correct uses of arithmetic and assignment operators
(e) Describe the correct uses of relational operators and logical operators
(f) Describe the use of sequence, selection and iteration

Outcome 3

Write a simple computer program.

Performance criteria

(a) Implement variables and data types in code
(b) Use naming conventions correctly
(c) Implement arithmetic and assignment operators in code
(d) Implement selection, relational operators and logical operators in code
(e) Implement sequence and iteration in code
(f) Combine sequences, selections and iterations to create a complete, working program, which accurately implements a simple algorithm

National Unit Specification: Statement of standards (cont)

Unit title: Computer Programming (SCQF level 4)

Evidence requirements for this unit

Learners will need to provide evidence to demonstrate the performance criteria across all outcomes.

Evidence is required to demonstrate that learners have achieved all outcomes and performance criteria. Assessors should use their professional judgement, subject knowledge and experience, and understanding of their learners to determine the most appropriate ways to generate evidence and the conditions and contexts in which they are used.

The evidence requirements for this unit will consist of **one** type of evidence: **product** evidence.

The product evidence will consist of **at least one** algorithm and **at least one** corresponding complete, working computer program written in a high-level programming language. The algorithm(s) and program(s) may be **simple** but **must include sequence, selection and iteration**.

The code must also demonstrate that learners can:

- use variables and naming conventions correctly.
- use at **least two** different data types.
- implement at **least two** different arithmetic operators.
- implement at **least two** assignment operators.
- implement at **least two** relational operators.
- implement at **least two** logical operators.
- implement iteration with at **least two** different types of loop.

The **product** evidence may be generated in supervised or unsupervised conditions with access to learning materials. 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.

The SCQF level (Level 4) of this unit provides additional context on the nature of the required evidence and the associated standards. The SCQF level descriptors should be used (explicitly or implicitly) when making judgements about the evidence.

The support notes provide specific examples of instruments of assessment that could be used to generate the required evidence (see *Guidelines on Approaches to Assessment*).

National Unit Support Notes

Unit title: Computer Programming (SCQF level 4)

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 40 hours.

Guidance on the content and context for this unit

The purpose of this unit is to introduce learners to problem solving using algorithms (computational thinking), as well as the key foundational concepts underpinning all programming languages. Learners will be given the opportunity to develop skills, knowledge and understanding in computational thinking, programming concepts, and implement those programming concepts.

This unit is intended for learners studying any type of computing (although not exclusively) and is particularly suitable for learners who want to progress to, or who are on, Computer Science courses or related courses, such as Software Development or Games Development. It is suitable for learners who have little or no experience of programming, but will also be valuable for those with previous experience of programming. It is also suitable for learners who want to know how to program for personal and/or vocational purposes, not just for academic reasons. It should be of particular interest to learners with an interest in STEM.

The unit covers the following knowledge and skills: algorithms, basic programming constructs (sequence, selection and iteration), variables, naming conventions, data types, arithmetic operators, assignment operators, relational operators and logical operators.

Although the focus is on acquiring programming skills that are transferrable and can apply to all programming languages, it is anticipated that learners will gain experience in at least one modern programming language and Integrated Development Environment (IDE).

The type of problems set for learners and the development environment they will use are at the discretion of the centre and will vary depending on the resources available to the centre. If the unit is delivered as part of an award, the problems set and development environment may reflect the subject area of the award and the emphasis individual centres have placed on that award. The chosen language could be Python, Basic, Java, C#, C++ and the IDE could be Visual Studio. However, any other modern programming language and environment, which make use of the programming concepts that must be covered, would be acceptable.

It is recommended that centres produce a brief for the learners, stating the problems to solve and the development environment or environments available to them. It is up to centres as to how prescriptive and specific the brief should be in terms of the problems set and the development environments they can make use of. It is anticipated that most centres will recommend a single development environment for learners to use.

National Unit Support Notes (cont)

Unit title: Computer Programming (SCQF level 4)

Please note that the following guidance, relating to specific outcomes, does not seek to explain each performance criterion, which is left to the professionalism of the teacher. It seeks to clarify the statement of standards where it is potentially ambiguous. It also focuses on non-apparent teaching and learning issues that may be over-looked, or not emphasised, during unit delivery. As such, it is not representative of the relative importance of each outcome or performance criterion.

Outcome 1

This outcome covers computational thinking by teaching learners how to create algorithms by breaking down routine problems into logical steps. These problems should be set at an appropriate level for students studying at SCQF level 4. Learners will learn how to break down a large problem into a series of smaller steps.

Outcomes 2 and 3

Outcome 2 covers foundational programming concepts, including variables, naming conventions, data types, arithmetic operators, assignment operators, relational operators, logical operators, sequence, selection and iteration. The same programming concepts are covered in Outcome 3, but whereas in Outcome 2 learners are expected to gain an understanding of these concepts, in Outcome 3 they must be able to implement them in at least one short program based on an algorithm.

Guidance on approaches to delivery of this unit

It is anticipated that learners would complete the outcomes sequentially, so they would complete Outcome 1, before moving onto Outcome 2, before finally moving onto Outcome 3. However, that does not mean that they should only cover the knowledge and skills required for each individual outcome before attempting the summative assessment for each outcome. It is anticipated that the knowledge and skills required for Outcome 1 will be covered first and that summative assessment may occur at that point before moving onto Outcome 2. However, it is expected that learners will cover the knowledge and skills required for Outcomes 2 and 3 before attempting the summative assessments for both of those outcomes.

For **Outcome 1,** learners should learn how to create algorithms to solve simple problems. This requires looking at how problems can be broken down into logical steps, which is computational thinking. The problems set do not need to be computing problems; in fact, initially, it would make sense to look at problems that they are familiar with, such as how to get from home to college, etc. They should also consider how large problems could be broken down into smaller steps.

For **Outcomes 2 and 3,** learners should learn about the programming concepts, techniques and skills mentioned in the content section. They should spend most of their time learning about each concept and then putting it into practice in the chosen programming language. Very simple algorithms should be provided to them, which they should be tasked with turning into code.

National Unit Support Notes (cont)

Unit title: Computer Programming (SCQF level 4)

A suggested distribution of time, across the outcomes, is:

Outcome 1: 12 hours
Outcome 2: 12 hours
Outcome 3: 16 hours

Summative assessment may be carried out at any time. However, when testing is used (see evidence requirements) it is recommended that this is carried out towards the end of the unit (but with sufficient time for remediation and re-assessment). When continuous assessment is used (such as the use of a web log), this could commence early in the life of the unit and be carried out throughout the life of the unit.

There are opportunities to carry out formative assessment at various stages in the unit. For example, formative assessment could be carried out on the completion of each outcome to ensure that learners have grasped the knowledge contained within it. This would provide assessors with an opportunity to diagnose misconceptions, and intervene to remedy them before progressing to the next outcome.

Guidance on approaches to assessment of this unit

Evidence can be generated using several types of assessment. The following are suggestions only. There may be other methods that would be more suitable to learners. Assessment evidence is required at all stages and outcomes. It must be documented and recorded electronically or in written/printed form; however, it is encouraged to look at alternate approaches making use of modern technology.

One approach to assessment is the use of a **practical exercise** as the instrument of assessment. This exercise would require learners to solve a simple problem by creating an algorithm and associated code in a high-level language. The resulting evidence could be a flowchart and a program listing.

If this approach is taken, it is recommended that the evidence is assessed using a checklist to ensure that minimum standards are achieved.

Suitable problems include:

- adding up numbers from 1 to n
- printing multiplication tables
- printing prime numbers
- sorting numbers
- guessing a secret number
- printing the next 20 leap years
- encoding text
- checking if a string is a valid email address
- checking if a word is a palindrome

The amount of control will vary from context to context. However, in every case, the conditions of assessment must be controlled to some extent. Where the amount of control is low, the amount of authentication should rise. It is not acceptable to produce evidence in lightly controlled conditions with little authentication.

National Unit Support Notes (cont)

Unit title: Computer Programming (SCQF level 4)

Authentication may take various forms including, but not limited to, oral questioning and plagiarism checks. Some forms of evidence generation (such as video recordings) have intrinsic authentication and would require no further means of verification. Where evidence is not generated under closely controlled conditions (for example, out of class), then a statement of authenticity should be provided by the learner to verify the work as their own, and state any necessary sources and permissions.

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.

Formative assessment could be used to assess learners' knowledge at various stages throughout the life of the unit. An ideal time to gauge their knowledge would be at the end of each outcome. This assessment could be delivered through an item bank of selected response questions, providing diagnostic feedback to learners.

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 will provide opportunities for learners to develop Core Skills in *Problem Solving* and *Information and Communication Technology* at SCQF Level 4.

The Processing Information component of *Information and Communication Technology* could be developed in Outcome 3, where the learner has to create a computer program.

The Critical Thinking and Planning and Organising components of *Problem Solving* could be developed in Outcome 1, where the learner has to create an algorithm to solve a problem. Aspects of *Problem Solving* could be developed further in Outcome 3, where the learner has to implement an algorithm into code.

History of changes to unit

Version	Description of change	Date

General information for learners

Unit title: Computer Programming (SCQF level 4)

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 provide you with foundational programming skills and a basic knowledge of the principles of computer programming.

This is a non-specialist unit, suitable if you want to know how to program for vocational purposes, and if you wish to appreciate programming for academic or personal purposes. It is particularly suitable if you have an interest in STEM.

You will gain a range of practical skills and acquire relevant underpinning knowledge. You will learn how to write basic code in a contemporary high-level language and appreciate basic programming techniques.

On completion of this unit, you will know how to write simple programs to solve real-world problems. You will be able to apply foundational programming concepts by implementing them in a programming environment.

You may progress to H2YC 45 *Computer Programming* at SCQF level 5.

Outcomes

On successful completion of the unit, you will be able to:

1 Write an algorithm to represent the solution to a simple problem.
2 Describe basic programming concepts.
3 Write a simple computer program.

In Outcome 1, you will cover computational thinking by learning how to create algorithms by breaking down simple problems into logical steps. You will also learn about basic programming constructs, such as sequence, selection and iteration.

In Outcome 2, you will learn about foundational programming concepts, including instructions, variables, naming conventions, data types, arithmetic operators, assignment operators, relational operators, logical operators, sequence, selection and iteration.

In Outcome 3, you will learn how to implement those programming concepts in short programs based on algorithms.