Next Generation Higher National Unit Specification

Software Development (SCQF level 7)

Unit code: J68V 47

SCQF level: 7 (16 SCQF credit points)

This unit is available in a restricted delivery model from academic session 2025

This unit specification provides detailed information about the unit to ensure consistent and transparent assessment year on year. It is for lecturers and assessors, and contains all the mandatory information you need to deliver and assess the unit.

Published: September 2025 (version 1.0)

Contents

Unit purpose

This unit introduces learners to the principles and practice of software development. It is a non-specialist unit, intended for learners studying HNC Computing or a related subject. It may also be appropriate for learners with an interest in any science, technology, engineering and mathematics (STEM) subject. No previous knowledge or experience of programming is required.

The unit covers software development concepts by allowing learners to build, test and deploy a software application. It examines the differences between software development life cycle models, and learners understand the importance of each development stage in producing robust applications. Learners explore practical programming concepts and gain hands-on experience of building software applications following a suitable design solution. They examine different testing strategies and understand the importance of testing, while developing software applications and preparing them for deployment.

The unit focuses on learners developing software individually. Developing software as part of a team is the focus of the Software Development unit at SCQF level 8.

On completion of the unit, learners have knowledge and skills using various software development methods, techniques and tools at an intermediate level, and can demonstrate them by analysing, designing, implementing, testing and deploying a software solution. This provides them with a good foundation to progress to the Software Development unit at SCQF level 8.

Unit outcomes

Learners who complete this unit can:

1. explain the software development life cycle
2. build a software application from a design
3. test a software application
4. deploy a software application

Evidence requirements

Learners must provide product evidence.

The product evidence relates to outcomes 1, 2, 3 and 4. Competence in outcome 1 is inferred from the product evidence.

Learners’ product evidence must demonstrate that they can:

- write a requirements specification
- design a software solution
- implement the solution in code
- carry out testing
- document the solution
- deploy the software

They must create at least one complete, working, error-free program. The problem should be non-trivial, with the focus on the software development life cycle rather than the complexity of code. The code must be sufficiently complex to illustrate:

- data structures
- control structures
- modular programming
- parameter passing
- program I/O
- software libraries

Learners must produce both maintenance and user documentation.

Learners can produce evidence over an extended period of time, in lightly-controlled conditions, with authentication.

The standard of evidence should be consistent with the SCQF level of the unit.

Knowledge and skills

Knowledge	Skills
Learners should understand: • the software development life cycle • the software development method • deliverables for each stage in the software development life cycle • good programming practice • software security • requirements analysis • software development tools and environments • software design, including algorithms • the syntax and semantics of a programming language • programming techniques, including structured programming • program control structures • parameter passing between modules • basic data structures • input and output file operations • testing strategies and methods used in software development • types of errors in software development • error handling • software documentation • software deployment process	Learners can: • create requirements specification • write algorithms • desk-check algorithms • create design documents • write code following a design • apply structured programming • use modular code • use arithmetic, logic and Boolean operators • use control constructs • use parameter passing to transfer values between modules • use standard libraries • use array data structures • use input/output file operations • use error handling techniques • create a test plan using a defined strategy • perform software testing • create test logs • create error logs • write maintenance documentation • write user documentation • deploy a software application

Meta-skills

You must give learners opportunities to develop their meta-skills throughout this unit. We have suggested how to incorporate the most relevant ones into the unit content, but you may find other opportunities.

Self-management

This includes focusing, integrity, adapting and initiative. The most relevant are:

- focusing:
  - paying attention
- adapting:
  - critical reflection
  - self-learning
- initiative:
  - independent thinking
  - self-motivation
  - taking responsibility

Social intelligence

This includes communicating, feeling, collaborating and leading. The most relevant are:

- communicating:
  - receiving information
  - listening
  - giving information

- leading:
  - leading others
  - being a change catalyst

Innovation

This includes curiosity, creativity, sense-making and critical thinking. The most relevant are:

- curiosity:
  - information sourcing
  - recognising problems
- creativity:
  - using imagination
  - visualising
  - having a maker mentality
- sense-making:
  - recognising patterns
  - holistic thinking
  - analysing
- critical thinking:
  - deconstruction
  - logical thinking
  - showing judgement
  - computational thinking

Literacies

This unit provides opportunities to develop the following literacies.

Numeracy

Learners develop numeracy skills through coding tasks. This allows them to develop small applications, such as profit loss calculators, conversion calculators or mathematical programs.

Communication

Learners develop communication skills through a range of activities, including the production of documentation and various design tools.

Digital

Learners develop digital skills and computer literacy throughout the duration and stages of the project.

Learning for Sustainability

Throughout this unit, you should encourage learners to develop their skills, knowledge and understanding of sustainability.

This includes:

- a general understanding of social, economic and environmental sustainability
- a general understanding of the United Nations Sustainable Development Goals (SDGs)
- a deeper understanding of subject-specific sustainability
- the confidence to apply the skills, knowledge, understanding and values they develop in the next stage of their life

Delivery of unit

This is an optional unit in HNC Computing. You can deliver it as a stand-alone unit, or partially integrate it with elements of the Database Design Fundamentals unit.

The notional time for delivery and assessment is 80 hours. The amount of time you allocate to each outcome is at your discretion. We suggest the following distribution of time, including assessment:

Outcome 1 —Explain the software development life cycle (10 hours)

Outcome 2 —Build a software application from a design (40 hours)

Outcome 3 —Test a software application (20 hours)

Outcome 4 —Deploy a software application (10 hours)

Additional guidance

The guidance in this section is not mandatory.

Content and context for this unit

The unit introduces learners to software development.

Approaches to delivery

When you address complex technological topics, you should use high-level terms and concepts, and teach terminology in a hands-on and problem-solving context throughout the unit.

Approaches to assessment

We recommend that you use a programming assignment for learners to produce the required evidence (see ‘Evidence requirements’ section). Alternatively, you could maintain a portfolio of code, produced over the life of the unit. If you use this approach, you must take care to ensure that learners satisfy all the evidence requirements.

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 must consider the needs of individual learners when planning learning experiences, selecting assessment methods or considering alternative evidence.

Guidance on assessment arrangements for disabled learners and those with additional support needs is available on the assessment arrangements web page.

Information for learners

Software Development (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 unit introduces you to the principles and practices of software development. It is a non-specialist unit that needs little or no previous knowledge or experience. It is particularly suitable if you are studying HNC Computing or a related subject. It may be appropriate for other qualifications that require a general knowledge of software development.

This unit gives you hands-on experience in using software development concepts, by allowing you to design, build, test and deploy software solutions. You learn about various software development life cycle models and the importance of each software development stage in producing robust software. You explore practical programming concepts and gain hands-on experience of building software applications following a suitable design solution. You learn to use different testing strategies and methodologies to ensure that your applications run error-free and are ready for deployment. Although the unit is primarily practical, it does contain a theoretical section. You are introduced to concepts and terminologies using real-world examples of software development solutions.

The assessment involves designing, writing, testing, documenting and deploying a software solution.

Meta-skills

Throughout this unit, you develop meta-skills that are useful for the computing sector.

Meta-skills are transferable behaviours and abilities that help you adapt and succeed in life, study and work. There are three categories of meta-skills: self-management, social intelligence and innovation.

Self-management

This includes focusing, integrity, adapting and initiative. The most relevant are:

- focusing:
  - paying attention
- adapting:
  - critical reflection
  - self-learning
- initiative:
  - independent thinking
  - self-motivation
  - taking responsibility

Social intelligence

This includes communicating, feeling, collaborating and leading. The most relevant are:

- communicating:
  - receiving information
  - listening
  - giving information

- leading:
    - leading others
    - being a change catalyst

Innovation

This includes curiosity, creativity, sense-making and critical thinking. The most relevant are:

- curiosity:
    - information sourcing
    - recognising problems
- creativity:
    - using imagination
    - visualising
    - having a maker mentality
- sense-making:
    - recognising patterns
    - holistic thinking
    - analysing
- critical thinking:
    - deconstruction
    - logical thinking
    - showing judgement
    - computational thinking

Learning for Sustainability

Throughout this unit, you develop skills, knowledge and understanding of sustainability.

You learn about social, economic and environmental sustainability principles and how they relate to the computing sector. You also develop an understanding of the United Nations Sustainable Development Goals.

Administrative information

Published: September 2025 (version 1.0)

Superclass: CB

History of changes

Version	Description of change	Date

Please check SQA's website to ensure you are using the most up-to-date version of this unit.