Higher National Unit specification

General information

Unit title: Software Development: Project (SCQF level 8)

Unit code: HA4K 35

Superclass: CB

Publication date: January 2016

Source: Scottish Qualifications Authority

Version: 01

Unit purpose

The purpose of this Unit is to apply skills and knowledge of software analysis, design, implementation and testing to produce a software product composed of multiple sub-programs. The Unit is aimed at learners seeking a role as an entry-level software developer.

Learners will determine the scope and plan the development of a software project, design the operation and interaction of its components, produce working code to meet the requirements, and test a completed solution to prove functional operation. This will involve using an object-oriented programming approach and choosing appropriate data structures and algorithms to build a functional product. Learners will collaborate with others to deliver the completed project.

The Unit is suitable for learners who have completed the Units *Software Development: Analysis and Design* (SCQF level 8) and *Software Development: Implementation and Testing* (level 8) or have prior learning or experience of software development. The Unit involves the practical application of skills, and will give learners exposure to the whole software development life-cycle.

On completion of the Unit, learners will be competent in the production of software products and may progress to the Unit *Software Development: Project* at SCQF level 9 or into an entry-level software development job.

Outcomes

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

1 Plan the development of a moderately complex software product.
2 Design the structure of a moderately complex software product using object-oriented programming techniques.
3 Develop a moderately complex software product using an object-oriented programming language.
4 Test the operation and acceptance of a moderately complex software product.

Higher National Unit Specification: General information (cont)

Unit title: Software Development: Project (SCQF level 8)

Credit points and level

2 Higher National Unit credits at SCQF level 8: (16 SCQF credit points at SCQF level 8)

Recommended entry to the Unit

The Unit is suitable for learners who have completed the Units *Software Development: Analysis and Design* (SCQF level 8) and *Software Development: Implementation and Testing* (level 8) or have prior learning or experience of software development. The Unit involves the practical application of skills, and will give learners exposure to the whole software development life-cycle.

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**.

Higher National Unit specification: Statement of standards

Unit title: Software Development: Project (SCQF level 8)

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.

Learners must collaborate with at least one other individual in one or more of the following Outcomes. For the purposes of definition in this Unit, a 'moderately-complex software product' is one that has multiple functionality and interfaces to APIs and data persistence.

Outcome 1

Plan the development of a moderately complex software product.

Knowledge and/or Skills

- Apply contemporary development approach
- Gather requirements information
- Prioritise requirements
- Validate acceptance criteria for product
- Outline test plan

Outcome 2

Design the structure of a moderately complex software product using object-oriented programming techniques.

Knowledge and/or Skills

- Produce wireframe designs
- Produce system interaction diagrams
- Produce object diagrams
- Write pseudocode
- Select algorithms

Higher National Unit specification: Statement of standards (cont)

Unit title: Software Development: Project (SCQF level 8)

Outcome 3

Develop a moderately complex software product using an object-oriented programming language.

Knowledge and/or Skills

- Build working software
- Accept user input
- Structure code idiomatically
- Apply object-oriented programming to meet design
- Process input according to design requirements
- Interact with data persistence
- Output results and feedback to user

Outcome 4

Test the operation and acceptance of a moderately complex software product.

Knowledge and/or Skills

- Check operation of code using a range of techniques
- Ensure acceptance criteria are met
- Measure coverage of tests
- Diagnose causes of errors
- Correct identified errors

Evidence Requirements for this Unit

Learners must collaborate with at least one other individual in one or more of the following Outcomes. For the purposes of definition in this Unit, a 'moderately-complex software product' is one that has multiple functionality and interfaces to APIs and data persistence.

Learners will need to provide evidence to demonstrate their Knowledge and/or Skills across all Outcomes. All evidence will be in the form of practical competence.

Evidence of practical competence will take the form of **at least one component** of a **moderately complex software product**. There is no requirement for separate evidence for cognitive and practical competence.

Higher National Unit specification: Statement of standards (cont)

Unit title: Software Development: Project (SCQF level 8)

The product must be sufficient to demonstrate the Knowledge and Skills in each Outcome. Candidates must develop **at least one component** of a **moderately complex software product**. Depending on its complexity, the product may or may not be error free. Moderately-complex products should be error free. More complex products may not. The following evidence is the **minimum** required:

1 Software development plan
2 Test plan
3 System design
4 Component designs
5 Source code
6 Object code
7 Test logs

This evidence may be supplied on paper or digitally or a combination of these.

The standard of the evidence should be consistent with the SCQF level of this Unit. At this level, the evidence should collectively or individually demonstrate:

- clear understanding of the overall software development process.
- computational thinking.
- specialist knowledge of the principles of programming.
- a discerning understanding of core theories, concepts, principles and terminology.
- using a range of professional skills, techniques, practices and/or materials associated with the software development industry, a few of which are advanced and/or complex.
- convey program design in well-structured and coherent form.
- take the lead on planning in familiar or defined contexts.
- work, under guidance, with others to acquire an understanding of current professional practice.

There are no **time limitations** on the production of evidence. The evidence may be produced at any time during the life of the Unit. Learners may use reference materials when undertaking assessment.

Evidence 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.

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

Higher National Unit Support Notes

Unit title: Software Development: Project (SCQF level 8)

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.

Guidance on the content and context for this Unit

This Unit may be delivered as a stand-alone Unit or following completion of the two other Units making up the Professional Development Award (PDA) in Software Development at SCQF level 8.

Learners should be encouraged to actively explore various hard-copy and internet resources including blog posts, tutorial guides and question and answer sites in order to familiarise themselves with a wide range of methodologies and tools as appropriate to each Learning Outcome. In addition, they must be aware of the internet safety, security, confidentiality and health and safety procedures of the organisation. It is also important to maintain the security and confidentiality of data and information and therefore leaners should be encouraged to back up and check for viruses on a regular basis in order to reduce the risk of loss of evidence of their achievement.

The overall aim of this Unit is for learners to apply skills and knowledge of software analysis, design, implementation and testing to produce a moderately-complex software product composed of multiple sub-programs. The focus of the Unit is on practical competencies and learners will analyse a problem, design the operation and interaction of program components, produce working code to meet requirements, and test a completed solution to prove functional operation, while working collaboratively on at least one aspect of the development life-cycle.

It is anticipated that substantial amount of time will be spent on practical tasks, and although learners may use tools that they have sourced themselves; appropriate tools to complete all of the Outcomes should be made available through their centre.

Outcome 1

This Outcome allows the learner to apply analysis and planning techniques produce a plan for developing a moderately-complex software product. They should demonstrate their understanding of an appropriate software development process, such as Agile or another contemporary methodology.

Learners are expected to produce planning documents for gathering information, prioritising requirements and creating acceptance test plans,

There is opportunity for learners to demonstrate collaboration with others, for example in the specification of features. The overall intention is that their work completed for Outcome 1 will be taken into design in Outcome 2.

Higher National Unit Support Notes

Unit title: Software Development: Project (SCQF level 8)

Outcome 2

The main purpose of this Outcome is for learners to design the structure of their product and the operation of their code for a moderately-complex software product based on object-oriented programming techniques.

Learners will demonstrate the visualisation of their product through design artefacts such as wireframes, system interaction and object diagrams. They will use pseudocode to demonstrate their choice of algorithms and code structure for the operation of their intended product

There is opportunity for learners to demonstrate collaboration with others, for example in the user experience design. The overall intention is that their planning will be taken into implementation in Outcome 3.

Outcome 3

This Outcome allows learners to develop a moderately-complex software product. Learners will turn the design of their product into a working software product using an object-oriented language and object-oriented design practices. Pseudocode from the design phase should be turned into well-structured working code using a combination of appropriate objects, constructs, sub-programs and functions to meet design requirements.

The product should interact with users and data storage to perform a minimum of two different functions. The product should be integrated with a minimum of one third-party API and demonstrate use of code libraries. User inputs should be validated against agreed criteria, and appropriate results/feedback outputted to the users.

The choice of object-oriented language is not specified, but learners should use idioms appropriate to their chosen language.

There is opportunity for learners to demonstrate collaboration with others, for example in using the third-party API or in data storage integration. The overall intention is that the product implementation in Outcome 3 will be used as a starting point for Outcome 4.

Outcome 4

This Outcome allows learners to apply their knowledge and skills to test a moderately-complex software product. Learners will demonstrate their ability to check the operation of code to ensure agreed criteria are met using a range of techniques. This may include testing against their own test plan, manual testing to a use-case plan, or automated testing using ‘Unit’ or ‘integration’ or other approaches to testing code.

Learners should demonstrate that they have completed a minimum of 50% testing of the core code structure of their product. Learners need to be able to decipher error messages and identify what file/function/line is the source of the error and then to suggest alternative solutions and correct the code accordingly.

Higher National Unit Support Notes

Unit title: Software Development: Project (SCQF level 8)

Learners could write their own test plan based on the work completed in the previous Outcomes. If learners cannot identify errors in their own code, eg in the product created to meet previous Outcomes, then they will need to be provided with errors in code of equal complexity to test against.

There is opportunity for learners to demonstrate collaboration with others, for example in end-user or operational testing.

Guidance on approaches to delivery of this Unit

This Unit should be delivered once learners have completed the *Analysis and Design* and *Implementation and Testing* Units at level 8. It would be expected that the delivery of this Unit's Outcomes progresses sequentially and that assessment would be undertaken at the end of each Outcome, eg learners should be confident in their analysis and design skills before completing implementation and then testing.

It is recommended that learners be guided through the full life-cycle of several increasingly complex software development projects as part of learning and teaching prior to assessment. This will allow learners to gain confidence in the application of their skills with more independence.

Throughout the Unit, the focus of projects could be on software products that have a relevance to the learner and to their further learning. Products could be anything from a command-line bank-account management app through to a web-based social network site.

Guidance on approaches to assessment of this Unit

Evidence Requirements for this Unit will be for practical competence. Evidence can be generated using different types of assessment. The following are suggestions only. There may be other methods that would be more suitable to learners.

The assessment for all Outcomes could comprise a series of practical tasks completed over a period of time. It is recommended that a holistic assessment covering all four Outcomes is used. Learners may have access to online resources, and the assessment need not be done under supervision, but all work should be authenticated as the learner's own. This could be done by oral questioning of the learner and/or observation of some aspects of the learner's work. For each Outcome, learners should devise, or be presented with, a product definition of an appropriately complex software product Learners should use the same product for all Outcomes, but alternative evidence may be used where this is not possible.

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.

Higher National Unit Support Notes (cont)

Unit title: Software Development: Project (SCQF level 8)

Outcome 1: Plan the development of a moderately-complex software product.

Learners may be presented with, or devise their own, moderately-complex product definition.

- **Apply contemporary development approach**

  Learners should demonstrate use of a contemporary software development methodology, such as Agile, to outline the development process for a moderately-complex software product.

- Gather and prioritise requirements information and produce acceptance criteria for product

  Learners should describe how the product will function for different types of users and journeys through the system. Their findings should be broken down into individual requirements, prioritised to indicate a sequence of implementation. Each requirement should include information about its acceptance criteria.

- **Test plan**

  Learners should prepare an acceptance test plan to meet agreed criteria for their product using appropriate software testing methodologies.

Assessment could take the form of a demonstration of a comprehensive project plan using a project management software tool, or a paper-based method such as index cards or post-it notes.

Outcome 2: Design the structure of a moderately-complex software product.

Learners may develop the product scoped in Outcome 1, or develop a product of equal complexity.

- **Produce wireframe designs**

  Wireframe designs should indicate both the expected user interface and user interaction, output and feedback. There should be appropriate wireframes for the major cases of functionality and for edge-case scenarios.

  Assessment could take the form of an 'elevator-pitch' style presentation, a written report format or be presented in a diagrammatic style.

- Produce system interaction diagrams

  Learners should be able to demonstrate the complete operation or flow-through the input/outputs of their product. Assessors should select two interactions between code and user input or output and ensure learners have produced detailed structure charts or system interaction diagrams for them.

  Assessment tools for this might take the form of white-boarding in a discussion, or sampling from a complete set of system interaction diagrams.

- **Produce object diagrams**

  Learners should be able to demonstrate that they have identified the classes, attributes and operations required for the product and the relationships between the objects.

  Assessment tools for this might take the form of CRC cards or Use Case diagrams for the major components of their implementation.

- **Write pseudocode and select algorithms**

  Learners should describe a minimum of two sub-routines of their design and one interaction of multiple objects by writing pseudocode outlining how the algorithms they have selected would function in the operational program. Learners' pseudocode should also demonstrate how it would be incorporated into a larger product if necessary.

  Assessment might take the form of observed discussion of the learner's pseudocode.

Outcome 3: Develop a moderately-complex software product.

Learners may develop the product scoped in Outcome 1 and designed in Outcome 2, or develop a product of equal complexity if this is not possible

- Build working software

  This is the core activity of this Outcome, so the primary measure of success is to produce working code. This means that the product runs from start to finish without exceptional errors. A minimum of 50% of this Outcome should be based on achieving working code.

  Code should be well-structured using an object-oriented programming approach, and be readable with its intention clearly documented with comments or idioms.

  Learner's product should interact with the user and process their input according to design requirements. Users should receive feedback from the product and be able to continue interaction. Results should be stored to enable persistent use of the software over time, ie users should be able to come back to find results of previous operations. Storage could take the form of file system, database or an appropriate alternative.

  Assessment might take the form of observation by assessor of the working product, with access to source code to check the structure for coverage of relevant knowledge and skills.

Higher National Unit Support Notes (cont)

Unit title: Software Development: Project (SCQF level 8)

Outcome 4: Test the operation and acceptance of a moderately-complex software product.

Learners may test the operation of the product they have developed for Outcome 3, or a product of equal complexity.

- **Check operation of code using a range of techniques**

- **Ensure acceptance criteria on test plan are met with adequate test coverage**

  Learners should be able to run their code from start to finish and check interaction with third-party APIs and data storage using a range of techniques. They should test code against their test plan to ensure an appropriate level of acceptance for a range of users.

- **Diagnose and correct errors**

  Learners should diagnose any identified errors and be able to rectify the error(s) to return to working code.

  This type of assessment may take the form of a professional discussion by Assessor with learner demonstrating the working code and explaining steps undertaken to diagnose and correct one or more errors in their own product.

  Alternatively, learners could be asked to write a 'bug report' on program code provided to them and asked to document the steps required to duplicate a bug in their product. This could be followed by learner making appropriate corrections.

  This type of assessment might take the form of a blog post, report or a personal demonstration to the assessor of what steps the learner took to achieve results.

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

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

The Unit provides opportunities for developing Computational Thinking skills.

History of changes to Unit

Version	Description of change	Date

General information for learners

Unit title: Software Development: Project (SCQF level 8)

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 apply skills and knowledge of software analysis, design, implementation and testing to produce a software product composed of multiple sub-programs. The Unit is aimed at learners seeking a role as an entry-level software developer.

You will determine the scope and plan the development of a software project, design the operation and interaction of its components, produce working code to meet the requirements, and test a completed solution to prove functional operation. This will involve using an object-oriented programming approach and choosing appropriate data structures and algorithms to build a functional product. You will collaborate with others to deliver the completed project.

The Unit is suitable for learners who have completed the Units *Software Development: Analysis and Design* (SCQF level 8) and *Software Development: Implementation and Testing* (level 8) or have prior learning or experience of software development. The Unit involves the practical application of skills, and will give you exposure to the whole software development life-cycle.

You will need to provide evidence to demonstrate your Knowledge and/or Skills across all Outcomes. Evidence of your cognitive and practical competence will take the form of at least one component of a moderately-complex software product.

On completion of the Unit, you will be competent in the production of software products and may progress to the Unit *Software Development: Project* at SCQF level 9, or into an entry-level software development job.