National Unit Specification: general information

UNIT **Software Development (Advanced Higher)**

NUMBER DF2Y 13

COURSE Computing (Advanced Higher)

SUMMARY

This Unit is designed to develop knowledge and understanding of software development and to develop practical skills in software development through the use of a high level language within an appropriate software development environment. In particular, it will consolidate and extend candidates' experience of standard language constructs and algorithms, and extend understanding of the software development process.

On completion of the Unit, the candidate should be able to apply this knowledge and understanding, and these skills to solve practical problems.

It is designed for candidates undertaking the Advanced Higher Computing Course, but it is also suitable for anyone wishing to extend and deepen their experience of software development beyond Higher level.

OUTCOMES

1. Demonstrate knowledge and understanding of the principles of software development, software development languages and environments, high level language constructs and standard algorithms.
2. Demonstrate practical skills in the context of software development using contemporary hardware and an appropriate software development environment.

Administrative Information

Superclass: CB

Publication date: April 2005

Source: Scottish Qualifications Authority

Version: 1

National Unit Specification: general information (cont)

UNIT Software Development (Advanced Higher)

RECOMMENDED ENTRY

While entry is at the discretion of the centre, candidates would normally be expected to have attained one of the following or equivalent:

- *Software Development* (Higher) Unit
- Higher Computing

CREDIT VALUE

1 credit at Advanced Higher (8 SCQF credit points at SCQF level 7*)

**SCQF credit points are used to allocate credit to qualifications in the Scottish Credit and Qualifications Framework (SCQF). Each qualification in the Framework is allocated a number of SCQF credit points at an SCQF level. There are 12 SCQF levels, ranging from Access 1 to Doctorates.*

CORE SKILLS

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

National Unit Specification: statement of standards

UNIT Software Development (Advanced Higher)

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 the Scottish Qualifications Authority.

OUTCOME 1

Demonstrate knowledge and understanding of the principles of software development, software development languages and environments, high level language constructs and standard algorithms.

Performance criteria

(a) A range of advanced computing terminology is used appropriately.
(b) Technically accurate descriptions and explanations are related to a range of familiar and unfamiliar contexts.
(c) Descriptions of high level language constructs and standard algorithms are correct.
(d) Conclusions, predictions and generalisations are made from knowledge and understanding.

Evidence requirements

Written or oral evidence that the candidate can describe and explain the principles of software development accurately and concisely. Evidence should be obtained using questions in a closed-book test, under supervision, lasting no more than 45 minutes. The test must sample across the range of the content (see Computing (Advanced Higher) Course content) in each of the following areas:

- the software development process
- software development languages and environments
- high level programming language constructs
- standard algorithms.

(The content statements are also reproduced for convenience as a table in the support notes for this Unit).

The standard to be applied is illustrated in the National Assessment Bank items available for this Unit.

If a centre wishes to design its own assessments for this Unit, they should be of a comparable standard.

National Unit Specification: statement of standards (cont)

UNIT Software Development (Advanced Higher)

OUTCOME 2
Demonstrate practical skills in the context of software development using contemporary hardware and an appropriate software development environment.

Performance criteria

(a) A wide range of appropriate hardware is used effectively and efficiently.
(b) A wide range of appropriate features of software is used effectively and efficiently.
(c) Practical tasks are planned and organised independently.
(d) Practical tasks are undertaken in an appropriate range of familiar and unfamiliar contexts.

Evidence requirements

Observation checklist showing that the candidate has carried out practical activities, demonstrating all of the following practical skills, as defined in the content statements (see Computing (Advanced Higher) Course content).

- analysis and design, with feasibility study
- design and creation of user-friendly interface
- comparison of two sorting or searching algorithms
- implementation of file handling
- use of module libraries
- testing of software
- producing user and technical documentation
- evaluating software.

Hard copy evidence should be provided of implementation and **two** other of these activities.

These practical skills may all be demonstrated in a single extended software development task, or in a number of smaller tasks.

The candidate will be allowed access to books, notes and on-line help while completing the task(s).

(The content statements are also reproduced for convenience as a table in the support notes for this Unit)

The standard to be applied is illustrated in the National Assessment Bank items available for this Unit.

If a centre wishes to design its own assessments for this Unit, they should be of a comparable standard.

National Unit Specification: support notes

UNIT Software Development (Advanced Higher)

This part of the Unit specification is offered as guidance. The support notes 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 content for this Unit is detailed below (and also in the National Course Specifications: Course details). Content statements in the left-hand column describe the content covered in the corresponding Unit at Higher level, and are included here to clarify the context for the new learning for this Unit. They indicate the prior learning required by the candidate before undertaking new learning within this Unit. Content in the right-hand column is the new content for this Unit.

Content Statements: Software development process	
Higher	Advanced Higher
Explanation of the iterative nature of the software development process. *Description of the purpose of the software specification as a legal contract.* *Explanation of the importance of each stage (analysis, design, implementation, testing, documentation, evaluation, maintenance) of the development process.*	Description of the progression through project proposal, feasibility study (economic, legal, technical and time), operational requirements document (ORD) and system specification, detailed design, implementation, component testing, system and acceptance testing, evaluation and maintenance.
Identification of the personnel involved at each stage (client, system analyst, project manager, programmer, independent test group) and brief descriptions of their roles.	
Description and exemplification of pseudocode and one graphical design notation structure diagram or other suitable) including data flow.	Comparison of different user-interface design styles (menu-driven, textual, graphical).
Description and exemplification of the top-down design and stepwise refinement.	
Explanation of the need for systematic and comprehensive testing.	Explanation of module, component and beta (acceptance) testing. Description of debugging techniques: dry runs, trace tables (tools), break points.
Explanation of the need for documentation at each stage.	
Evaluation of software in terms of robustness, reliability, portability, efficiency and maintainability.	
Description and exemplification of corrective, adaptive and perfective maintenance.	

National Unit Specification: support notes (cont)

UNIT Software Development (Advanced Higher)

Content Statements: Software development languages and environments	
Higher	Advanced Higher
Description and comparison of procedural, declarative and event-driven languages.	Description of object-oriented language. Comparison of object-oriented with procedural, declarative, event-driven and low level languages. Explanation of the trends in language development (low level to high level, 4th generation).
Comparison of the functions, uses and efficiency of compilers and interpreters.	
Description of the features and uses of scripting language (including creating and editing a macro). Explanation of the need for and benefits of scripting languages.	
Description of the use of module libraries.	Description of the use and advantages of Computer-Aided Software Engineering (CASE) tools

Content Statements: High level programming language constructs	
Higher	Advanced Higher
Description and exemplification of the following constructs in pseudocode and an appropriate high level language: ♦ *string operations (concatenation and substrings)* ♦ *formatting of I/O* ♦ *CASE (or equivalent multiple outcome selection)*	Description and exemplification of the following constructs in pseudocode and an appropriate high level language: ♦ file handling: write, read, delete item, create new file, open, read, and close file
Description and exemplification of real, integer and boolean variables and 1-D arrays	Description and exemplification of the following variable types/data structures: 2-D arrays, records, queues, stacks.
Description and exemplification of procedures/subroutines, user-defined functions, modularity, parameter passing (in, out, in/out), callpass by reference/value, local and global variables, scope.	Description and exemplification of ♦ user-defined module libraries

National Unit Specification: support notes (cont)

UNIT Software Development (Advanced Higher)

Content Statements: Standard algorithms	
Higher	Advanced Higher
Description and exemplification of the following standard algorithms in pseudocode and an appropriate high level language: ♦ *linear search* ♦ *counting occurrences* ♦ *finding min/max*	Description and exemplification of the following standard algorithm in pseudocode and an appropriate high level language: ♦ binary search Simple description and comparison of sort algorithms in terms of number of comparisons and use of memory: ♦ selection sort using two lists ♦ simple sort ♦ bubble sort
	Simple description and comparison of linear and binary search algorithms

National Unit Specification: support notes (cont)

UNIT Software Development (Advanced Higher)

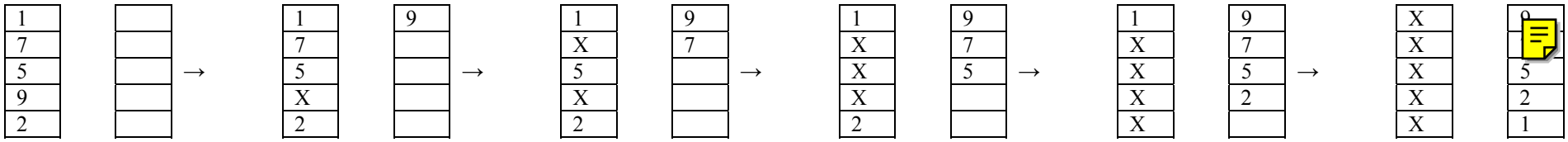
Sort algorithms

The three sort algorithms required for this Unit are as follows:

Selection sort using two lists

Set up new empty list
    For number of items in list
    Find max in unsorted list
    Transfer it to first empty element of new list
    Replace with dummy data in old list
Next

Two list sort



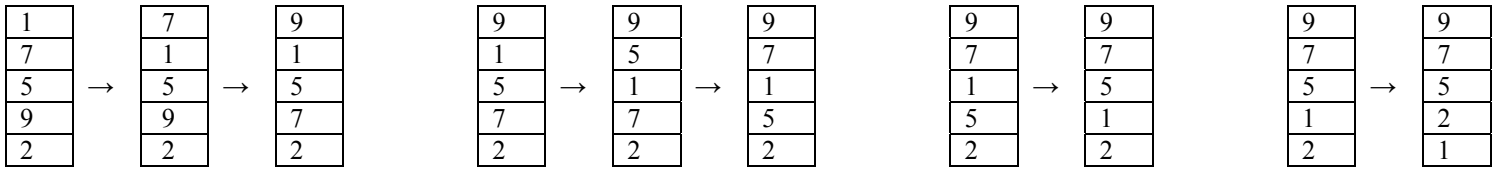
National Unit Specification: support notes (cont)

UNIT Software Development (Advanced Higher)

Simple Sort

Start with item 1
If item 2 > item 1, swap
If item 3 > item 2, swap
And so on to the end of the list
Repeat process with item 2
And so on to the end of the list

Simple sort



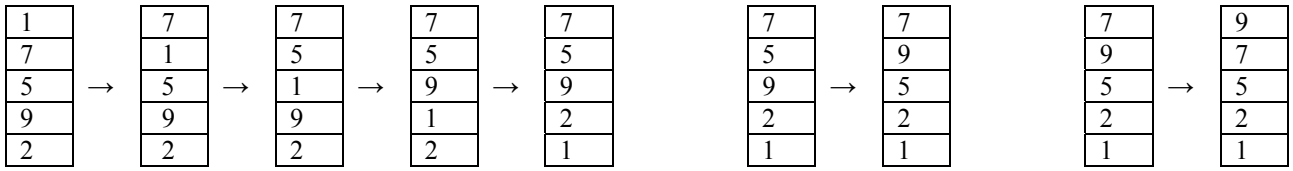
National Unit Specification: support notes (cont)

UNIT Software Development (Advanced Higher)

Bubble Sort

Start with item 1
If item 2 > item 1, swap
If item 3 > item 1, swap
And so on to the end of the list
Repeat process until no more swaps

Bubble sort



Animation web site

Animations of the simple sort and bubble sort can be found at:

http://www.cs.brockport.edu/cs/javasort.html

National Unit Specification: support notes (cont)

UNIT Software Development (Advanced Higher)

GUIDANCE ON LEARNING AND TEACHING APPROACHES FOR THIS UNIT

Candidates will require individual access to appropriate computer hardware and an appropriate software development environment throughout this Unit. While the learning will usually be achieved in the context of a single software development environment, students will benefit from having some experience of alternative software development environments.

The two Outcomes should be delivered in an integrated way rather than sequentially. For Outcome 2, the practical activities should be taught and used to illustrate and exemplify the knowledge and understanding required for Outcome 1. These practical activities can be used to generate evidence for Outcome 2.

Candidates who have completed the *Software Development* Unit at Higher level should already have covered the content listed in the left-hand column of the content statement grids, but may need to revise this material before progressing to the right hand column.

The main purpose of this Unit is to consolidate and extend candidates' experience of standard language constructs and algorithms, and to extend an understanding of the software development process. For those candidates undertaking the Computing Course, this will provide a suitable basis for undertaking the '*Developing a Software Solution*' Unit and the Coursework Project.

The amount of time spent on each content area will vary depending on the teaching methodology used and the ability and prior experience of the candidates. However, the following times are suggested as a rough guide:

software development process	4 hours
languages and environments	5 hours
language constructs	12 hours
standard algorithms	15 hours

1½ hours should be set aside to:

- administer the Outcome 1 test
- gather evidence for Outcome 2

A further 2 ½ hours is allowed for remediation and re-assessment if required.

If the Unit is delivered as part of a Course, the Course documentation will provide further information on teaching and learning in a Course context, including the identification of a number of 'themes' to facilitate holistic learning across the Course.

National Unit Specification: support notes (cont)

UNIT Software Development (Advanced Higher)

GUIDANCE ON APPROACHES TO ASSESSMENT FOR THIS UNIT

National Assessment Bank tests have been created specifically to assess Outcome 1 of the Unit. This assessment consists of a closed book test, and must be conducted under supervision. In order to gain success in this Outcome, the candidate must achieve at least the cut-off score for the test. If a centre wishes to design its own assessments for this Unit, they should be of a comparable standard.

Outcome 2 requires the candidate to demonstrate practical skills while developing software using an appropriate high-level language environment. These practical skills may all be demonstrated in a single extended software development task, or in a number of smaller tasks. The skills will normally be demonstrated by the candidate during the teaching and learning activities in the Unit, rather than as separate formal assessment activities. The candidate will be allowed access to books, notes and on-line help while completing the task(s).

To gain success in this Outcome, the candidate must demonstrate practical skills in the following contexts and at an appropriate level as defined by the content statements (see Computing (Advanced Higher) Course content):

- analysis and design (including feasibility study)
- design and creation of a user-friendly interface
- comparison of two sorting or searching algorithms
- implementation of file handling
- use of module libraries
- testing of software
- producing user and technical documentation
- evaluating software

A pro-forma observation checklist for Outcome 2 is provided in the National Assessment Bank materials.

Hard copy evidence is required of implementation and **two** other of these skills; this need not be formal documentation - it could include hand-written notes on design, hard copy of coding, or screen shots demonstrating implementation and/or testing.

All evidence must be retained by the centre. The assessment of this Unit is subject to moderation by SQA.

National Unit Specification: support notes (cont)

UNIT Software Development (Advanced Higher)

CANDIDATES WITH ADDITIONAL SUPPORT NEEDS

This Unit Specification is intended to ensure that there are no artificial barriers to learning or assessment. The additional support needs of individual candidates should be taken into account when planning learning experiences, selecting assessment instruments or considering alternative Outcomes for Units. For information on these, please refer to the document *Guidance on Assessment Arrangements for Candidates with Disabilities and/or Additional Support Needs* (SQA, 2004).