National Unit Specification: general information

UNIT Software Development (Higher)

NUMBER DF2Y 12

COURSE Computing (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 develop and consolidate candidates' familiarity with standard language constructs and algorithms, and enable candidates to develop modular programs combining standard constructs. 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 Higher Computing Course, but is also suitable for anyone wishing to extend and deepen their experience of software development beyond Intermediate 2 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.

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:

- Intermediate 2 Software Development Unit
- Intermediate 2 Computing
- Standard Grade Computing Studies at Credit Level

Administrative Information

Superclass:	CB
Publication date:	April 2004
Source:	Scottish Qualifications Authority
Version:	01

Additional copies of this Unit Specification can be purchased from the Scottish Qualifications Authority. The cost for each Unit Specification is £2.50. (A handling charge of £1.95 will apply to all orders for priced items.)

National Unit Specification: general information (cont)

UNIT Software Development (Higher)

CREDIT VALUE
1 credit at Higher (6 SCQF credit points at SCQF level 6*).

**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 Skill components in this Unit.

National Unit Specification: statement of standards

UNIT Software Development (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 computing terminology is used appropriately.
b) Technically accurate descriptions and explanations are related to practical and familiar 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 content (see Computing (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 (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 range of appropriate hardware is used effectively and efficiently.
b) A range of appropriate features of software is used effectively and efficiently.
c) Practical tasks are planned and organised with minimal guidance.
d) Practical tasks are undertaken in an appropriate range of familiar 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 (Higher) Course Content).

- analysis and design
- implementation of two appropriate language constructs
- implementation of two standard algorithms
- implementation of parameter passing
- testing of software
- producing documentation
- evaluating software

Hard copy evidence should be provided of implementation and one other of these skills.

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 online 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 (Higher)

This part of the Unit Specification is offered as guidance.

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 Intermediate 2 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	
Intermediate 2	Higher
Description of the stages (in order) of the software development process: analysis, design, implementation, testing, documentation, evaluation, maintenance.	Explanation of the iterative nature of the software development process. Description of the purposes of the software specification and its status as a legal contract. Explanation of the importance of each stage (analysis, design, implementation, testing, documentation, evaluation, maintenance) of the development process.
	Identification of the personnel at each stage (client, systems analyst, project manager, programmer, independent test group) and brief description of their roles.
Description and exemplification of pseudocode and one graphical design notation (structure diagram or other suitable).	Description and exemplification of pseudocode and one graphical design notation (structure diagram or other suitable) including data flow.
	Description and exemplification of top-down design and stepwise refinement.
Description and exemplification of appropriate test data (normal, extreme and exceptional).	Explanation of the need for systematic and comprehensive testing.
Description of the features of a user guide and a technical guide.	Explanation of the need for documentation at each stage.
Evaluation of software in terms of fitness for purpose, user interface and readability.	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 (Higher)

Content Statements: Software development languages and environments	
Intermediate 2	Higher
Description and comparison of machine code and high level languages.	Description and comparison of procedural, declarative and event-driven languages.
Explanation of the need for translation; Description of the functioning of interpreters and compilers.	Comparison of the functions, uses and efficiency of compilers and interpreters.
Description of the process of recording a macro and assigning it to a keystroke; Description of examples of the use of macros.	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 features and use of a text editor.	Description of the use of module libraries.

Content Statements: High level programming language constructs	
Intermediate 2	Higher
Description and exemplification of the following constructs in pseudocode and an appropriate high level language: ♦ *input and output, assignment, arithmetical operations (+,-,*,/,^) and logical operators (AND, OR, NOT)* ♦ *fixed loops, conditional loops, simple and complex conditions, conditional statements (IF), nested loops*	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 numeric and string variables and 1-D arrays.	Description and exemplification of real, integer and boolean variables and 1-D arrays.
Description and exemplification of pre-defined functions.	Description and exemplification of procedures/subroutines, user-defined functions, modularity, parameter passing (in, out, in/out), call by reference/value, local and global variables, scope.

National Unit Specification: support notes (cont)

UNIT Software Development (Higher)

Content Statements: Standard Algorithms	
Intermediate 2	Higher
Description and exemplification of the following standard algorithm in an appropriate high level language: ♦ *input validation*	Description and exemplification of the following standard algorithms in pseudocode and an appropriate high level language: ♦ linear search ♦ counting occurrences ♦ finding min/max
Recognition of appropriate use of the following standard algorithms: ♦ *input validation* ♦ *find min/max,* ♦ *count occurrences,* ♦ *linear search*	

National Unit Specification: support notes (cont)

UNIT Software Development (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, candidates 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 Intermediate 2 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 develop and consolidate candidates' experience of standard language constructs and algorithms, and develop these into extended modular programs.

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	6 hours
language constructs	14 hours
standard algorithms	12 hours

1½ hours should be set aside to:

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

UNIT Software Development (Higher)

A further 2½ 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.

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 examination conditions. 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 of the Unit, rather than as separate formal assessment activities. The candidate will be allowed access to books, notes and online 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 (Higher) Course content):

- analysis and design (including data flows)
- implementation of two language constructs
- implementation of two standard algorithms
- implementation of parameter passing and subroutines
- 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 one 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 (Higher)

SPECIAL NEEDS

This Unit Specification is intended to ensure that there are no artificial barriers to learning or assessment. Special needs of individual candidates should be taken into account when planning learning experiences, selecting assessment instruments or considering special alternative Outcomes for Units. For information on these, please refer to the SQA document *Guidance on Special Assessment Arrangements* (SQA, September, 2003).