

Software Development: Linked Data Structures: An Introduction

HIGHER NATIONAL DIPLOMA

软件开发：链式数据结构（初级）

【英】苏格兰学历管理委员会（SQA）

Unit Student Guide

COMPUTING: Software Development

DM32 35



 中国时代经济出版社


SCOTTISH
QUALIFICATIONS
AUTHORITY

Software Development: Linked Data Structures: An Introduction

HIGHER NATIONAL DIPLOMA

软件开发：链式数据结构（初级）


【英】苏格兰学历管理委员会 (SQA)

Unit Student Guide

COMPUTING: Software Development

江苏工业学院图书馆
藏书章

M31.35

 中国时代经济出版社

SCOTTISH
QUALIFICATIONS
AUTHORITY



著作权合同登记 图字：01-2005-3760号

图书在版编目 (CIP) 数据

软件开发：链式数据结构. 初级/苏格兰学历管理委员会著. -北京：中国时代经济出版社，2005.8

ISBN 7-80169-918-1

I. 软… II. 苏… III. ①软件开发-教材-英文 ②数据结构-教材-英文 IV. TP311.52

中国版本图书馆CIP数据核字 (2005) 第048208号

“First published by CMEPH”

“All Rights Reserved”

“Authorized Apograph/ Translation/Adaptation of the editions by the Scottish Qualifications Authority. All Intellectual Property Rights vest in the Scottish Qualifications Authority and no part of these “Works” may be reproduced in any form without the express written permission of Scottish Qualifications Authority”

Software Development: Linked Data Structures: An Introduction

软件开发：链式数据结构（初级）

苏格兰学历管理委员会著

出版者	中国时代经济出版社
地址	北京市东城区东四十条24号 青蓝大厦东办公区11层
邮政编码	100007
电话	(010) 68320825 (发行部) (010) 88361317 (邮购)
传真	(010) 68320634
发行	各地新华书店
印刷	北京鑫海达印刷有限公司
开本	787×1092 1/16
版次	2005年8月第1版
印次	2005年8月第1次印刷
印张	11.75
定价	30.00元
书号	ISBN 7-80169-918-1/G·257

版权所有 侵权必究

Contents

1	Introduction to the Scottish Qualifications Authority	1
2	Introduction to the Unit	3
2.1	What is the Purpose of this Unit?	3
2.2	What are the Outcomes of this Unit?	3
2.3	What do I Need to be Able to do in Order to Achieve this Unit?	4
2.4	Approximate Study Time for This Unit	8
2.5	Equipment/Material Required for this Unit	9
2.6	Symbols Used in this Unit	9
3	Assessment Information for this Unit	13
3.1	What Do I Have to Do to Achieve This Unit?	13
4	Suggested Lesson Plan	15
5	Learning Material	17
5.1	Section 1	18

5.2 Section 2	62
6 Additional Reading Material	143
7 Solutions to Activities	145
8 Copyright References	155
9 Acknowledgements	157
Appendix 1 Unit Specification	159

1

Introduction to the Scottish Qualifications Authority

This Unit **DM32 35 Software Development: Linked Data Structures** has been devised and developed by the Scottish Qualifications Authority (SQA). Here is an explanation of the SQA and its work:

The SQA is the national body in Scotland responsible for the development, accreditation, assessment, and certification of qualifications other than degrees.

Its website can be found at: www.sqa.org.uk.

The SQA's functions are to:

- devise, develop and validate qualifications, and keep them under review
- accredit qualifications
- approve education and training establishments as being suitable for entering people for these qualifications
- arrange for, assist in, and carry out the assessment of people taking SQA

qualifications

- **quality-assure education and training establishments that offer SQA qualifications**
- **issue certificates to candidates.**

In order to pass SQA units, students must complete prescribed assessments. These assessments must meet certain standards.

The Unit Specification outlines the five Outcomes that students must complete in order to achieve this Unit. The Specification also details the knowledge and/or skills required to achieve the Outcomes. The Evidence Requirements prescribe the type, standard and amount of evidence required for each Outcome.

2

Introduction to the Unit

2.1

What is the Purpose of this Unit?

This Unit is designed to enable candidates to become familiar with abstract data types and the linked data structures used to implement them within software systems. This knowledge will be supplemented by research, analysis, design and coding of structures in order to create applications to meet user requirements.

2.2

What are the Outcomes of this Unit?

The five Outcomes of the Unit are to:

1. Identify and describe the concepts of dynamic memory allocation and linked data structures, explain their advantages over array data structures
2. Define interfaces and design necessary algorithms for abstract data types
3. Implement software that makes effective use of linked data structures
4. Design and implement software that makes effective use of a tree abstract data type

5. Manipulate a nested data structure in an application .

Further details can be found in Appendix 1—Unit Specifications.

2.3

What do I
Need to be
Able to do in
Order to
Achieve this
Unit?

You need to achieve each of the individual learning Outcomes. Achievement of the first Outcome requires the candidate to correctly answer 60% or more of the questions in a multiple choice test. The test will consist of 20 questions and will cover areas such as:

- Abstract data types including lists, trees, queues and stacks
- Dynamic memory allocations and the concepts of node, data element and pointer/link
- The advantages and disadvantages of linked data structures over array structures.

Outcome 2 requires that you develop and provide the interface definitions, algorithms and perform a desk check for two different abstract data structures. The abstract data structure to be developed will be selected from:

- A Stack
- A Queue

- **A List**

For each of the selected data structures you need to provide a brief description of its nature and fully define the interface of the data structure, to include algorithms, pre-conditions and post-conditions. The algorithms must be data independent and the following functions are required as a minimum:

- **Initialise**
- **Is empty**
- **Add item**
- **Remove item**

Note: There may be other algorithms required in the development of your structures.

A desk check will be used to ensure that you understand the algorithms you have developed.

Outcome 3 requires you to implement the interfaces defined in Outcome 2. You will write and test the code utilised to implement the two interfaces you have developed. This will include a written test strategy and written test plan with appropriate and relevant test data. This will involve the production of two working models. Each of the working models must have the following interface definitions working:

- Initialise
- Is empty
- Add item
- Remove item

Note: You may also develop other interface operations.

Outcome 4 requires you to look at the tree abstract data structure. You will design and implement the interface using recursive algorithms. You will write, document and test the code for a binary search tree. The working model will cover the following minimum operations:

- Initialise
- Is empty
- Add item
- Remove item
- Tree traversal (in order, post-order and pre-order).

You will also produce a written test strategy and written test plan with appropriate and relevant test data.

Outcome 5 requires the production of an application that makes use of two or more of the data structures that you have designed and implemented. You will be supplied with a program brief, test harness and a test plan. You will produce a working model and this must be tested using the supplied test harness and test plan.

2.4
Approximate
Study Time
for This Unit

It is anticipated that you will receive approximately 80 hours of tuition for this Unit, the time spent according to the following schedule:

Outcome 1: 8–10 hours

Outcome 2: 8–10 hours

Outcome 3: 10–15 hours

Outcome 4: 20–25 hours

Outcome 5: 18–20 hours

You will also need approximately 80 hours of individual study time in addition to the time noted above. Developing software is a time-consuming activity, with a wide scope for frustrating delays, and this additional time is given as guidance.

2.5 Equipment/ Material Required for this Unit

In order to complete this Unit students will need access to a computer with a programming language translator. The material presented in this Unit should be suitable for use with a wide range of imperative programming languages, but will assume the use of the Java programming language where actual program code samples are provided.

You will need to complete the following Unit Student Guides to meet all the requirements of this unit:

- Unit Student Guide—Software Development: Linked Data Structures: An Introduction
- Unit Student Guide—Software Development: Linked Data Structures: Advanced

2.6 Symbols Used in this Unit

The various Learning Materials sections are designed so that you can work at your own pace, with tutor support. As you work through the Learning Materials (see Section 5), you will encounter symbols. These symbols indicate that you are expected to complete a task. **These tasks are not Outcome Assessments.** They are exercises designed to consolidate learning or encourage thought, in preparation for the Outcome Assessment (see Section 3—Assessment Information for this Unit).

Activity



This symbol indicates an Activity. Usually, Activities are used to improve or consolidate your understanding of the subject in general or a particular feature of it.

For this Unit, you are asked to undertake eight consolidation Activities. 7 of the Activities are included Unit Student Guide—Software Development: Linked Data Structures: An Introduction. A further one Activity is included in Unit Student Guide—Software Development: Linked Data Structures: Advanced.

Everything is provided for you to check your own responses. Answers to the Activities are found at the back of the Unit Student Guide. Where suggested responses to Activities are provided in the Unit Student Guide, **students are strongly discouraged from looking at these responses before they attempt the Activity.** The Activities throughout the Unit Student Guide will help you to prepare yourself for the formal assessments, and to identify topic areas in which you might require clarification and additional tutor support. The Activities will not serve this purpose if you look at the answers before trying the Activities!

Activities are designed to be checked by you. No tutor input is necessary at this stage unless special help is requested, although from time to time your tutor may wish to view your responses to Activities to see how you are progressing.

