Unit 74: Principles of Software Design and Development

Unit code:	**K/600/0121**
NQF Level 3:	**BTEC National**
Credit value:	**10**
Guided learning hours:	**60**

Aim and purpose

This unit focuses on the design and development process as a whole and would be an appropriate way to introduce learners to programming before they undertake more focused programming language units. The unit examines the business context within which programming solutions can be developed.

Unit introduction

A comprehensive range of generic application software and utilities are available to businesses across many different areas of employment. Examples include spreadsheets and databases, as well as more focused packages, such as payroll or computer aided design (CAD). Sometimes a business identifies that cannot be addressed in this way a need these situations it is necessary to build a solution using an appropriate computer language.

In order to develop a programmed software solution which meets business and user needs, it is necessary to understand the problem and be very clear on the user requirements. Problems are often caused by poor understanding of user need, as well as poor planning. A wide range of different programming languages are available, each with quite different characteristics and features. Although, learners will be focusing on one particular language, they will build an appreciation of why different high-level languages are available and why they are chosen in particular situations.

Learners will be made aware of the documenting and testing required as an integral part of creating a computer program. A major part of learners' time will be spent on familiarising themselves with fundamental software development processes and concepts. This will give learners a firm foundation from which to tackle the more focused programming units.

Learning outcomes

On completion of this unit a learner should:

1 Know the nature and features of programming languages

2 Be able to use software design and development tools

3 Be able to design a program

4 Be able to implement a programmed solution.

Unit content

1 Know the nature and features of programming languages

Types of language: procedural languages; object-oriented; visual languages; other languages eg script and markup languages; simple overviews and uses

Reasons for choice of language: organisational policy; suitability in terms of available features and tools; availability of trained staff; reliability; development and maintenance costs; expandability

Features: variables eg naming conventions, local and global variables, arrays; loops eg conditional (pre-check, post-check), fixed; conditional statements; case statements; logical operators; assignment statements; input statements; output statements

Data types: text; integer; floating point; byte; date; Boolean; other eg char, smallint; benefits of appropriate choice of data type eg additional validation, efficiency of storage

2 Be able to use software design and development tools

Software development life cycle: eg determination of scope, requirements gathering and specification, design, code, test, maintain

Design tools: as appropriate to problem eg structure diagrams, DFDs, ERM

Software structures: as appropriate to language chosen eg iteration, decisions, modules, functions, procedures, classes and objects

3 Be able to design a program

Requirements specification: inputs, outputs, processing, user interface; constraints eg hardware platforms, timescales for development

Design: as related to specific design technique chosen

4 Be able to implement a programmed solution

Testing and debugging: test strategy; test plan structure eg test, date, expected result, actual result, corrective action; error messages; specialist software tools eg debug

Technical documentation: requirements specification; other as appropriate to language eg form design, flowcharts, pseudocode, structured English, action charts, data dictionary, class and instance diagrams

User documentation: eg details of hardware platform required, loading instructions, user guide; getting help

Review: against specifications requirements; interim reviews

Assessment and grading criteria

In order to pass this unit, the evidence that the learner presents for assessment needs to demonstrate that they can meet all of the learning outcomes for the unit. The criteria for a pass grade describe the level of achievement required to pass this unit.

Assessment and grading criteria					
To achieve a pass grade the evidence must show that the learner is able to:		**To achieve a merit grade the evidence must show that, in addition to the pass criteria, the learner is able to:**		**To achieve a distinction grade the evidence must show that, in addition to the pass and merit criteria, the learner is able to:**	
P1	describe, using examples, why different types of programming languages have been developed [IE]	**M1**	illustrate the features of programming languages	**D1**	demonstrate use of a range of design and development tools
P2	describe, with examples, the benefits of having a variety of data types available to the programmer [CT, RL]	**M2**	justify the choice of data types used in a programmed solution	**D2**	evaluate a programmed solution.
P3	write and test the functionality of a number of internally documented programs that demonstrate the features available in a given language [SM]	**M3**	adapt and improve a programmed solution.		
P4	implement a working programmed solution [SM]				
P5	test a programmed solution [IE]				
P6	document a programmed solution.				

PLTS: This summary references where applicable, in the square brackets, the elements of the personal, learning and thinking skills applicable in the pass criteria. It identifies opportunities for learners to demonstrate effective application of the referenced elements of the skills.

Key	IE – independent enquirers	RL – reflective learners	SM – self-managers
	CT – creative thinkers	TW – team workers	EP – effective participators

Essential guidance for tutors

Delivery

It is possible that some learners may have existing programming experience, possibly gained through the BTEC First programme. It is therefore advised that some diagnostics be performed to establish the existing knowledge and skills, of learners. Structured workshops with supporting materials and access to tutors can be a very effective way to deliver this unit.

It is however likely that more formal input will be required from tutors, covering the range of content.

Learners will need to become familiar with the features and concepts used by the particular programming language taught learnt and undertake numerous exercises to become confident in their use.

Learners will concentrate on one particular programming language but need to have an awareness of other languages and the reasons for using different languages in different situations.

The software development cycle should be constantly referred to as the structure for developing programmed solutions. The various design tools such as data flow diagrams, may have been introduced to learners in other units, for examples. Note that these are given as examples of design tools and the tools chosen for learning should be appropriate to the task.

Learners will design, develop, test and document programs and sufficient time must be allowed for revisiting the procedures for testing and modifying programs retesting after modification and documenting the solutions both for the technician and the user.

Outline learning plan

The outline learning plan has been included in this unit as guidance and can be used in conjunction with the programme of suggested assignments.

The outline learning plan demonstrates one way of planning the delivery and assessment of this unit.

Topic and suggested assignments/activities and/assessment
Introduction to unit. Programming languages – type, when used, why used. Assessing current level of knowledge. Mixture of tutor led discussions, practical exercises.
Programming features. Introducing features. Introducing data types. Mixture of tutor led demonstration and learner exercises, repeated over a number of sessions.
Assignment 1: Which Language?
Development life cycle. Structures and tools. Determining the requirement. Mixture of tutor led instruction, directed learning, practical exercises.
Assignment 2: Good Design!
Testing. Debugging, test strategy, documenting testing. Revision as required and practical work.
Documenting. User documentation (content). Technical documentation (content). Mixture of discussion of exemplar material, research, practical.
Evaluating. Revision of evaluation techniques.
Assignment 3: Problem Solved
Review of unit and assessment.

Assessment

It is suggested that this unit is, such as these given three assignments assessed through in the Programme of suggested assignments (PSA) table below.

As identified in the unit abstract, one key theme that should be emphasised through this unit, and the qualification, is that of creating solutions to meet defined requirements and user need. The evidence for P3 to P6 must be based on requirements relating to a particular situation. There should be a defined client and, ideally, this should not be the tutor. If circumstances dictate that no external client can be identified and a group assignment is set, then the scenario should be richly detailed and the requirements clearly specified in order for appropriate testing and review to take place.

For the first part of the assessment, learners could be asked to lead a session on programming languages, to be delivered to a group of trainee programmers. The evidence for P1 and M1 could be provided through a presentation. The presentations do not need to be actually presented verbally if sufficient detail or speakers' notes are provided. There are four different types of languages identified in the content of learning outcome 1, so learners are expected to respond with four different languages for P1. For M1, at least four features should be discussed and their use illustrated with examples.

A written response may be appropriate for P2 or a table with three or four columns. Alternatively, learners could use to present their evidence data type, example, space occupied and comment. The data types available may be different according to which language is chosen, however there is an expectation that learners know of the wider range of data types available. Six is noted as being a sufficient number in the unit content.

The context for this assignment will depend on the case study used. As noted above, P3 should be based on a substantial activity. The design should include the original requirement and include use of at least one design tool such as a structure diagram. For D1, learners will have used a range of design tools to document their design eg DFD, structured English, data dictionary. Evidence for M2 can be included with the design documentation and based on the data types used in the design.

This is the bulk of the practical work and evidence will come from the documentation (screen shots, coding etc) and a witness statement confirming the application works within defined boundaries. What functionality is required for the program to be termed 'working' should be clearly defined. This should include at least the basic functionality required by the user. The evidence for P4, P5 and P6 is likely to be a combination of program code, with user and technical documentation, test plans and results.

M3 is likely to relate to P5 and the evidence presented could be records of reviews and changes made. A second review after the changes have been made should confirm the improvement. Two improvements would be sufficient user interface, space used, speed of operation, for example additional input checking, increased facility to develop the code later on, etc.

The evaluation for D2 should include evaluation against the original requirement, user reviews and suggestions for further improvements.

Programme of suggested assignments

The table below shows a programme of suggested assignments that cover the pass, merit and distinction criteria in the assessment and grading grid. This is for guidance and it is recommended that centres either write their own assignments or adapt any Edexcel assignments to meet local needs and resources.

Criteria covered	Assignment title	Scenario	Assessment method
P1, P2, M1	**Assignment 1:** Which Language?	A programmer leads a session on programming languages for a group of trainee programmers.	Presentation. Handouts. Notes.
P3, M2, D1	**Assignment 2:** Good Design!	A programmer for a small business has been asked to design a program to meet a particular need.	Design documentation.
P4, P5, P6, M3, D2	**Assignment 3:** Problem Solved	A programmer implements and documents a design and ensures it meets the client's needs.	Test plan. Witness statement. User reviews. User guide. Technical documentation. Evaluation.

Links to National Occupational Standards, other BTEC units, other BTEC qualifications and other relevant units and qualifications

This unit forms part of the BTEC Art and Design sector suite. This unit has particular links with the following unit titles in the BTEC Art and Design suite:

Level 1	Level 2	Level 3
Introduction to Creative Use of Computers	Working With Digital Art and Design Briefs	Image Manipulation Using Computer Applications
	Working with Moving Image Briefs	3D Computer Modelling
	Working with Interactive Media Briefs	Digital Image Capture and Editing
	Computers in Graphic Design	

National Occupational Standards

This unit also provides development opportunities for some of the underpinning skills, knowledge and understanding of the following National Occupational Standards:

CCSkills Sector Skills Council

Design (revisions in draft form June 2009)

DES7 Contribute to the production of prototypes, models, mock-ups, samples or test pieces

DES23 Create 2D Designs using a Computer Aided Design System

DES24 Create 3D Models using a Computer Aided Design System

DES36 Develop and extend your design skills and practices

Skillset Sector Skills Council

Interactive Media and Computer Games

IM1 Work Effectively in Interactive Media

IM6 Use Authoring Tools to Create Interactive Media Products

Design for the Moving Image

DMI 1 Assist With The Technical Design Process

DMI 3 Contribute To The Production Of Designs Using IT

DMI 4 Assess The Technical Implications Of The Design Brief

Essential resources

Learners will need individual access to a particular programming language and development environment. They will also need a more limited access, possibly by demonstration, to other different types of languages.

Indicative reading for learners

Textbooks

Glassborow F – *You Can Do It: A Beginner's Introduction to Computer Programming* (John Wiley, 2003) ISBN 978-0470863985

Knuth D – *The Art of Computer Programming: Volumes 1-3, 2nd Edition* (Addison Wesley, 1999) ISBN 978-0201485417

Knuth D – *The Art of Computer Programming: Fascicles 0-4 v. 4* (Addison Wesley, 2009) ISBN 978-0321637130

Todd D – *Game Design: From Blue Sky to Green Light* (A K Peters, 2007) ISBN 978-1568813189

Wender K – *Cognition and Computer Programming* (Intellect Books, 1995) ISBN 978-1567500950

Journals

Computing and IT week

Computer Art Magazine

Creative Review

Design Magazine

Design Week

Journal of Information Technology

Websites

www.adobe.com	digital media software
bubl.ac.uk/link/c/computerprogramming-c++.htm	resources for C++
www.computerarts.co.uk	tutorials and examples of digital media
www.digitmag.co.uk	links and news on digital media
www.nsead.org/ict/index.aspx	case studies, examples of practice and links, related to new media and education in art and design

Delivery of personal, learning and thinking skills

The table below identifies the opportunities for personal, learning and thinking skills (PLTS) that have been included within the pass assessment criteria of this unit.

Skill	When learners are …
Independent enquirers	describing the development of programming languages
Creative thinkers	exploring the uses of different data types
Reflective learners	considering the strengths and weaknesses of different data types
Self-managers	implementing and testing a program.

Although PLTS are identified within this unit as an inherent part of the assessment criteria, there are further opportunities to develop a range of PLTS through various approaches to teaching and learning.

Skill	When learners are …
Independent enquirers	planning and carrying out research on programming languages
Creative thinkers	trying out alternative ways of developing programs to meet specific needs
Reflective learners	adapting and improving a programmed solution based on formal testing and review
Self-managers	writing and testing the functionality of a number of internally documented programs that demonstrate the features available in a given language
Effective participators	documenting, testing and reviewing a programmed solution.

Functional skills – Level 2

Skill	When learners are …
ICT – Use ICT systems	
Select, interact with and use ICT systems independently for a complex task to meet a variety of needs	designing and testing a program for a specific need
Use ICT to effectively plan work and evaluate the effectiveness of the ICT system they have used	implementing IT systems and business platforms appropriately to justify using computers in art and design
Follow and understand the need for safety and security practices	understanding security in using programming languages
Troubleshoot	debugging a program
ICT – Find and select information	
Access, search for, select and use ICT-based information and evaluate its fitness for purpose	evaluating source code for a specific purpose describing with examples the benefits of having a variety of data types available to the programmer
Mathematics	
Interpret and communicate solutions to practical problems in familiar and unfamiliar routine contexts and situations	designing and creating a working programmed solution based on a defined set of requirements
English	
Speaking and listening – make a range of contributions to discussions and make effective presentations in a wide range of contexts	describing using examples why different types of programming languages have been developed.