MODULE SPECIFICATION

Part 1: Information			
Module Title	Object Oriented Software Design and Development I		
Module Code	UFCFUM-15-2	Level	Level 5
For implementation from	2019-20		
UWE Credit Rating	15	ECTS Credit Rating	7.5
Faculty	Faculty of Environment & Technology	Field	Computer Science and Creative Technologies
Department	FET Dept of Computer Sci & Creative Tech		
Module type:	Standard		
Pre-requisites	None		
Excluded Combinations	None		
Co- requisites	None		
Module Entry requirements	None		

Part 2: Description

Overview: The purpose of this topic is to introduce the apprentices to the fundamental concepts of systems development through programming, computational thinking and data structures. They will analyse models of application development so that they can understand the key processes related to building functioning applications and appreciate the complexity of application development.

Educational Aims: Apprentices will learn the basic concepts of software design, data structures, programming, problem solving, programming logic, and fundamental software design techniques. This will include a review of traditional and contemporary software development methods including agile development. They will develop a holistic view of software engineering practice including gathering requirements, designing a solution, implementing a solution in a programming language, testing the completed application and deploying the solution to end users.

Outline Syllabus: Using an industry recognised language apprentices will:

Demonstrate an understanding of object-oriented concepts (e.g. classes, objects, inheritance, polymorphism, encapsulation)

Perform object-oriented analysis and design. This will need to incorporate valid object-oriented designs (e.g. use case, user stories)

Object-oriented Program Development, to a defined business' requirement

Software artefacts

Apply good business practice in all areas of the development life cycle

The use of an appropriate object oriented testing facility and tracking to be able to debug created program code to understand and rectify problems within the code (e.g. white box,black box, unit testing)

Security appraisal (e.g. attack risks, mitigation, planned changes)

Teaching and Learning Methods: Introductory lectures are supported by seminars, case studies, visits and practical workshops. In addition this module will be supported by interactive forums and learning tools.

150 hours study time of which 36 hours will represent scheduled learning.

Independent learning includes hours engaged with essential reading, case study preparation, assignment preparation and completion. Apprentice study time will be organised each week with a series of both essential and further readings and preparation for practical workshops.

This unit is practically based and designed to ensure that apprentices understand and develop their skills in advanced programming techniques. Apprentices will use the object-oriented facilities within C++ as a vehicle for this.

36 hours scheduled learning, 114 hours research, independent study and preparation for assessment work.

Scheduled learning will typically include lectures, seminars, supervision, external visits and an interactive forum.

All apprentices are expected to attend a series of tutorials.

Part 3: Assessment

This module is assessed by a combination of techniques: an examination (3 hours) and a practical build.

Component A – Exam

Apprentices will be required to sit a 3-hour exam that will require knowledge of the following object-oriented techniques, classes, objects, inheritance, polymorphism and encapsulation.

Apprentices will be required to perform object-oriented design using a taught methodology, against a business specification. They will need to employ various designs such as use case and user stories.

Apprentices will be required to create a sound program using a recognised object-oriented language from the designs that they have previously created. Apprentices must identify and apply best practices and standards throughout their practical build.

Component B – Practical Build

Apprentices will be given a business specification from which they will produce a solution. They will need to design their systems and apply their knowledge of the development lifecycle models to create a sound system.

The task will include development, implementing, testing and debugging. The testing of the program will need to be robust and thorough, using techniques such as white and black box testing. The program must be fully object-oriented to ensure a sound understanding of the benefits that are associated with object-oriented programming. After the testing, apprentices will be required to modify their existing code base and apply new fixes or modules

from their requirements and testing. The program will need to be fully documented and conform to industry standards.

Opportunities for formative assessment exist for the assessment strategy used. Verbal feedback and written feedback is given to all apprentices providing a personal platform for improvement.

First Sit Components	**Final Assessment**	**Element weighting**	**Description**
Set Exercise - Component B		50 %	Design, implement, test and correct a problem specification
Examination - Component A	✓	50 %	Examination (3 hours)
Resit Components	**Final Assessment**	**Element weighting**	**Description**
Set Exercise - Component B		50 %	Design, implement, test and correct a problem specification
Examination - Component A	✓	50 %	Examination (3 hours)

Part 4: Teaching and Learning Methods

Learning Outcomes

On successful completion of this module students will achieve the following learning outcomes:

Module Learning Outcomes	**Reference**
Describe the theory and concepts of the object-oriented paradigm	MO1
Understand software design approaches and patterns and can interpret and implement a given design	MO2
Analyse business and technical requirements and select appropriate solutions	MO3
Create analysis artefacts, such as Use Cases and/or User Stories	MO4
Design, implement, test, and debug software to meet a requirements specification	MO5
Develop moderately complex software solutions and software modifications to specified requirements	MO6
Debug own code and understand structure of programmes in order to identify and resolve issues	MO7
Identify and apply best practices and standards	MO8

Contact Hours

Independent Study Hours:	
Independent study/self-guided study	114
Total Independent Study Hours:	114
Scheduled Learning and Teaching Hours:	
Face-to-face learning	36

	Total Scheduled Learning and Teaching Hours:	36
	Hours to be allocated	150
	Allocated Hours	150
Reading List	*The reading list for this module can be accessed via the following link:* https://uwe.rl.talis.com/index.html	

Part 5: Contributes Towards
This module contributes towards the following programmes of study: