Unit 20: Advanced Programming

Unit code	**Y/615/1651**
Unit level	**5**
Credit value	**15**

Introduction

Features of programming languages that are considered advanced are used to develop software that is efficient; it can affect the performance of an application as well as the readability and extensibility of the code, improving productivity and therefore reducing cost. Many commercial applications available today, whether for productivity or entertainment, will have used one or more design pattern in their development. A design pattern is a description of how to solve a problem that can be used in many different situations and can help deepen the understanding of object-orientated programming and help improve software design and reusability.

The aim of this unit is to familiarise students with these features and their best practices to ensure that their code is in line with industry standards.

Among the topics included in this unit are: object-orientated programming; polymorphism, encapsulation, class aggregation/association, constructors/destructors, inheritance, abstract classes, interfaces, containers, generics, introduction to design patterns and Unified Modelling Language (UML).

On successful completion of this unit students will be able to write code in an object-orientated fashion using design patterns where necessary and be able to model their code structure in UML class diagrams. As a result they will develop skills such as communication literacy, critical thinking, analysis, reasoning and interpretation, which are crucial for gaining employment and developing academic competence.

Learning Outcomes

By the end of this unit students will be able to:

LO1. Examine the key components related to the object-orientated programming paradigm, analysing design pattern types.

LO2. Design a series of UML class diagrams.

LO3. Implement code applying design patterns.

LO4 Investigate scenarios with respect to design patterns.

Essential Content

LO1 **Examine the key components related to the object-orientated programming paradigm, analysing design pattern types**

Outline the object-orientated paradigm characteristics:

Encapsulation, polymorphism, constructors/destructors, sub objects, abstract/concrete, interface, method redefinition, generics/templates, containers.

Object-orientated class relationships:

Generalisation/inheritance, realisation, dependency, aggregation, composition.

Design patterns:

Creational, structural and behavioural.

LO2 **Design a series of UML class diagrams**

UML class design:

Analyse a code scenario and utilise a suitable UML tool to develop class diagrams.

LO3 **Implement code applying design patterns**

Implementation:

Using an appropriate language & IDE to develop code that implements design patterns and utilises techniques to produce secure code.

LO4 **Investigate scenarios with respect to design patterns**

Review the usage of design patterns:

Relating design patterns to a range of given scenarios