Module: Programming 152

Module name:	Programming 152
Code:	PRG152
NQF level:	5
Type:	Core – Diploma in Information Technology (all stream)
Contact time:	56 hours
Structured time:	8 hours
Self-directed time:	66 hours
Notional hours:	130 hours
Credits:	13
Prerequisites:	PRG151

Purpose

The aim of this course is to expand on the mastered knowledge obtained from Programming 151. This module will broaden the students programming skills base taking into account collections, abstract data types, parameterised methods and enumeration.

Outcomes

Upon successful completion of this module, the student will be able to:

- Demonstrate an informed understanding of built-in data types, collections, enumeration and practices and an informed understanding of the key terms, concepts, facts, general principles, rules and theories in the programming domains.
- Select and apply standard methods, procedures or techniques within the OOP software domain and scope of this module, and plan and manage an implementation and testing process in orde to develop an application.
- Identify, evaluate and develop an user-friendly interface in either console or Windows forms, including multiple forms.
- Identify, evaluate and solve defined, routine and advanced problems within the programming context,
- Apply optimal solutions based on relevant evidence and procedures or other forms of explanation appropriate to programming languages, discipline or practice, demonstrating an understanding of the consequences.
- Gather information from a range of sources, including oral or written, to select information appropriate to the task, and to apply basic processes of analysis, synthesis and evaluation on that information to determine its suitability as an optimal solution.

Assessment

- Continuous evaluation of work through 1 assignment.
- Continuous evaluation of work through formative tests and a summative test which assess the theoretical knowledge.
- Continuous evaluation of two projects, whereby the student must evaluate and present results on given problems.
- Final assessment through a written examination.

Teaching and Learning

Learning materials

Lecturer hand-outs and samples.

Prescribed Material

- Programming: Introduction – IT without frontiers series.

Additional Reference Material

- Sprankle, M., Hubbard, J. (2011). *Problem Solving and Programming Concepts* (9th Edition). Pearson.[ISBN-13: 978-0132492645]

Learning activities

The teaching and learning activities are an amalgamation of formal lectures based on theoretical concepts and practical guided instructions where the actual implementation can be practised. One mandatory assignment and two projects must be completed during the course. These practical components also serve to indicate areas of discussion that are needed to further flesh out some concepts.

Notional learning hours

Activity	Units	Contact Time	Structured Time	Self-Directed Time
Lecture		40.0		24.0
Formative feedback		9.0		
Project	2	7.0		15.0
Assignment	1			3.0
Test	3		6.0	11.0
Exam	1		2.0	13.0
		56.0	**8.0**	**66.0**

Syllabus

Algorithms

- Create algorithms to solve a computational problem.
- Explain how programs implement algorithms in terms of instruction processing, program execution, and running processes.
- Evaluate various built-in types for the chosen programming language to facilitate algorithm design.

Built-in data types

Abstract data Types

Introdution to Methods

Parameterised methods

Introduction to events

- Event arguments
- Observer design patterns

Enumeration

Use appropriate abstractions to facilitate writing programs: collections, procedures, application programming interfaces, and libraries.

Collections

- Queue
- Stack
- List

Functions

- Basic types of function
- Declaration and definition
- Function call
- Types of function
- Parameter passing

Introduction to User interface design

- Types of interface
- Introduction to interface design in Visual Studio

Introduction to architecture of Windows Forms

Windows forms basics

- Controls and containers

Introduction to events

- Delegates and events
- Event handling in Windows Forms applications

Testing and maintenance

- Assertions
- Unit testing
- Code refactoring