Module: Programming 151

Module name:	Programming 151
Code:	PRG151
NQF level:	5
Type:	Core – Diploma in Information Technology (all stream)
Contact time:	30 hours
Structured time:	6 hours
Self-directed time:	34 hours
Notional hours:	70 hours
Credits:	7
Prerequisites:	None

Purpose

The purpose of this module is to understand how software has helped people solve problems. The student will learn several general concepts that will allow them to formulate an understanding of a problem and develop an algorithm to support the solution. They will be introduced to arithmetic used in programming, sequences, selection and iteration control structures with built in data types. The student will be introduced to the concepts of Object Orienet Programming (OOP) and be able to apply their problem solving techniques and translate pseudocode into console programmes.

Outcomes

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

- Demonstrate a fundamental understanding of problem solving concepts, binary logic, flowcharts, pseudo code, built in datatypes and algorithms found within the software domain.
- Select and apply standard problem solving techniques within the software discipline, and to plan and manage an implementation process applied to problem solving.
- Identify, evaluate and solve defined arithmetic problems, and to apply solutions based on relevant evidence and understanding the consequences if an algorithm is poorly designed within a computer system.
- The ability to read, understand and solve problems; this includes the ability to design pseudocode to solve a given problem and to express it in a structured diagram.
- Operate in a range of familiar problem domains, demonstrating an understanding of different kinds of problems to be solved, their constituent parts and the relationships between these parts, and to understand how algorithms in one area impact on other areas within the same software system.

Assessment

- Continuous evaluation of work through 2 assignments.
- Continuous evaluation of work through formative tests and summative test which assesses the theoretical knowledge.
- 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: 9780132492645]

Learning activities

The teaching and learning activities consist of a combination of formal lectures on theoretical and practical concepts, exercises and discussions. Two mandatory assignments must be completed during the course. The experiences and progress on these practical components form the content of class discussions.

Notional learning hours

Activity	Units	Contact Time	Structured Time	Self-Directed Time
Lecture		27.0		13.0
Formative feedback		3.0		
Project				
Assignment	2			6.0
Test	2		4.0	8.0
Exam	1		2.0	7.0
		30.0	**6.0**	**34.0**

Syllabus

Overview of Programming

- Definition of a computer
- What is Programming?
- How do we write a program?
  - Problem-Solving Phase
  - Implementation Phase
  - Maintenance Phase

Algorithms

- Introduction to algorithms
- Describe why and how algorithms solve computational problems.

Problem Solving

- Problem Solving Techniques
  - Solve by Analogy
  - Means-Ends Analysis

- Divide and Conquer
- The Building-Block Approach
- Merging Solutions

Truth Tables and Logic Gates

- Logical Operations
- Switching Circuits and Binary Logic
- Logic Gates
- Hierarchy of Operations
- Boolean Expressions

Pseudocode

- What is Pseudocode?
- Why is Pseudocode useful?
- How to write pseudocode?
  - Rules
  - Example of Pseudocode

Flowcharts

- Flowchart Basics
- Programming with Flowcharts
- Pseudocode vs. Flowcharting
- Control Constructs
  - Variables and Assignment
  - Decision making
  - Loops structure
  - Combining Constructs

Introduction to OOP

- Characteristics of OOP
- Advantages and disadvantages of OOP
- Comparison between structured programming and OOP
- OOP Concepts

Introduction to .NET platform

- Introduction to .NET framework to create a C # programs

Introduction data types, variables and constants

- Built-in data types

Introduction to decision making structures and loop statements

- if statement
- if-else statement
- switch statement
- while loop
- do-while loop
- for loop

Creation of console menu driven applications