Liverpool John Moores University

Title:	MATHEMATICAL COMPUTER PROGRAMMING
Status:	Definitive
Code:	**4012MATHS** (120290)
Version Start Date:	01-08-2016
Owning School/Faculty:	Applied Mathematics
Teaching School/Faculty:	Applied Mathematics

Team	Leader
Steven Webb	Y
Ian Malabar	

Academic Level:	FHEQ4	**Credit Value:**	24	**Total Delivered Hours:**	72
Total Learning Hours:	240	**Private Study:**	168		

Delivery Options
Course typically offered: Standard Year Long

Component	Contact Hours
Lecture	24
Practical	48

Grading Basis: 40 %

Assessment Details

Category	Short Description	Description	Weighting (%)	Exam Duration
Technology	AS1	Series of programming lab tasks.	40	
Artefacts	AS2	Development of a piece of software.	60	

Aims

To develop IT problem solving skills
To become familiar with a range of mathematical programming techniques To gain an understanding of how software is developed
To prepare students for mathematical software development at higher levels, both work and study

Learning Outcomes

After completing the module the student should be able to:

1 Apply knowledge of computer programming constructs and algorithms to IT problems.
2 Demonstrate problem solving skills to create simple software solutions.
3 Evaluate alternatives and make sound judgements about data structures.
4 Investigate development environment tools for use in software development.
5 Demonstrate familiarity with using mathematical functions within programs.

Learning Outcomes of Assessments

The assessment item list is assessed via the learning outcomes listed:

Programming tasks	1	2	
Software development	3	4	5

Outline Syllabus

Computers and Computer Programming
-How programs work within computers
-Current programming languages and their evolution
-Programming cycle
-Interpreted, managed and native code
-Overview of methodologies: Structured / Imperative, Functional and OO
-Strong / Weak Typing

IDE
-Working with code
-Compiling, profiling, testing and organising code

Basic elements of programs -Syntax
-Variables/Types
-Expressions
-Input/Output and Devices
-Classes and methods

Control structures
-Conditionals / selection
-Loops / repetition
-Logical problem solving
-User defined classes/ADTs
-Value and Reference Types
-Arrays / Collections
-String manipulation
-Code structure, procedures/methods, callbacks.

-Recursion

Graphics
- Plotting graphs and statistical data

Learning Activities

Lectures – to introduce the programming theories and techniques Lab exercises – programs for students to write and test.

Further exercises – practical examples for students to work on in their own time
Directed reading – background reading to enable the lab work to be completed.

Notes

This module aims to develop programming and problem solving skills in students to help prepare them for work in mathematics and statistics.