MODULE **DESCRIPTOR**

MODULE TITLE	ALGORITHMS AND DATA STRUCTURES		
MODULE CODE	CO1406 (L4)	**CREDIT VALUE**	20 UK CREDITS / 10 ECTS
SCHOOL	SCHOOL OF SCIENCES		

MODULE **AIMS**

This module develops students' problem-solving skills while introducing common algorithms and data structures. Discussion of advantages and disadvantages of a limited range of algorithms and data structures will enable students to solve simple problems for themselves and to select and apply more complex published techniques. Implementation of algorithms and data structures will enhance students' programming skills and reinforce their programming language knowledge. Their programming toolkit will be extended to include dynamic memory allocation, pointers and the use of existing algorithms and data structures from a library.

- To develop problem-solving skills
- To evaluate common data structures
- To analyse the performance of common algorithms
- To enhance practical programming skills.

MODULE **CONTENT**

Indicative syllabus content:

Problem-solving
The problem-solving process: Identification, investigation, and problem definition, refining objectives, classification, generation of potential solutions, evaluation.
Divide and conquer, functional decomposition, search, greedy algorithms
Diagrams in problem-solving
Evaluating a solution: identifying and assessing criteria. Functional and performance testing.
Applications: the analysis and solution of problems to illustrate the effective use of data structures and algorithms (e.g. the implementation of sets using a variety of data structures)

Algorithms
Searching: Sequential Search, Binary Search, Simple Hashing Sorting: Selection Sort, Insertion Sort, MergeSort, QuickSort Recursion (e.g. Towers of Hanoi, mazes, knight's tour, 8-queens)

Programming Language Features
Pointers and dynamic memory allocation

Common Data Structures
Arrays, Lists, Queues, Trees

Using Implemented Algorithms and Data Structures
The use of simple container classes & iterators from an industrial-strength library (e.g. STL)

INTENDED **LEARNING OUTCOMES**

On successful completion of this module a student will be able to:

1. Generate, explain and evaluate potential solutions to a range of problems
2. Describe and evaluate common data structures and algorithms
3. Analyse the time and space complexity of simple algorithms
4. Select, justify and implement an appropriate data structure for a particular problem
5. Design algorithms to solve simple problems

TEACHING METHODS

Although much of the content of this module is a discussion of common algorithms and data structures, the key aim is the development of general problem-solving skills. The algorithms and data structures provide a toolkit that can be applied to solve problems, but the teaching approach is not simply to review these tools, but to help students to analyse a problem, suggest and evaluate several solutions and justify the chosen solution. To allow students to focus on problem-solving rather than technology and to encourage discussion, some problems will be abstract: what is the fastest way to identify a poisoned drink from 127 glasses of wine? How can you detect a duplicate ticket number at a sporting event? More technical problems will be introduced and addressed in a programming language-independent way: illustrated and explained using pseudocode to encourage students to think first and code later. Students will implement and test algorithms and data structures using a high-level language, reinforcing their understanding and enhancing their programming. New skills will include the use of pointers and dynamic memory allocation along with an introduction to the use of existing algorithms and data structures from a commercial class library. Lectures will be used to introduce new information and concepts, to walk through algorithms and the operation of data structures, and to review solutions developed in tutorials or implemented in practicals. Tutorials will be used for group and individual problem-solving. There will be an emphasis on diagrams, walkthroughs and discussion of solutions and alternatives. Practicals will be used to implement and evaluate algorithms or data structures based around graduated problem sheets with a mix of small exercises and extended problems.

ASSESSMENT METHODS

This module is assessed through a portfolio of lab exercises (covering implementation, analysis of performance and discussion of alternatives) and in-class quizzes.