Module Code	**CIS2344**
Module Title	**ALGORITHMS, PROCESSES AND DATA**
School(s) Involved in Delievery	Computing and Engineering
Name(s) of Course(s)	FDSc Computer Games Technology (Programming) BSc Software Development BSc Software Development with... BSc Internet Systems Development BSc Secure & Forensic Computing BSc Computer Games Programming BSc Computing BSc Computer Science MEng Software Engineering BEng Computer Systems Engineering
Module Leader	Hugh Osborne
Location	Queensgate
Module Type	Core
Credit Rating	20 credits
Level	Intermediate
Learning Methods	**Supervised Learning:** 44 hrs **Unsupervised Learning:** 156 hrs
Prerequisites	None
Recommended Prior Study	Knowledge of an object oriented programming language (e.g. JAVA), or equivalent
Corequisites	None
Professional Body Requirements	None
Graded or Non-Graded	Graded
Barred Combinations	None

Synopsis

This module exposes students to both a range of classic algorithms and data structures and encourages them to develop well structured, appropriate and efficient algorithms and data structures using both sequential and concurrent paradigms.

Outline Syllabus

- Complexity and Computability, algorithm equivalence, NP completeness -
  - Notations to express algorithm efficiency, e.g. Big O notation.
- Standard algorithms and data structures -
  - Linear data structures, trees and graph structures, construction sorting and searching.
  - Recursion - implementation issues.
- Basic concepts and features of concurrency -
  - Semaphores, shared memory, message passing, monitors, threads.
- Modelling concurrent process behaviour -
  - Deadlock, starvation, non-determinism.
- Implementation -
  - Programming language semantics for concurrent process implementation.

Learning Outcomes:

1. Knowledge and Understanding

   On completion of this module the student will be able to:

   1. Discuss the classification of algorithms according to efficiency and complexity.
   2. Describe a range of useful algorithms and the properties and uses of common data structures.
   3. Demonstrate a knowledge of the characteristics of a range of concurrency paradigms.
2. Abilities

   On completion of this module the student will be able to:-

   1. Develop complex algorithms and data structures to solve practical problems.
   2. Use a standard notation to analyse the efficiency and complexity of algorithms.
   3. Use a standard notation to express concurrent process behaviour.
   4. Implement working concurrent software using appropriate programming language semantics.

Assessment Strategy

Formative assessment for this module will take the form of weekly exercises with appropriate feedback, including model solutions.

Additionally, a software artifact will be developed, with accompanying documentation, that will be submitted to a full assessment, in the context of Learning Outcomes 1.1, 1.2, 2.1 and 2.2, and returned with feedback.

Summative assessment will consist of one in-course assignment.

The in-course assignment will involve the development of algorithms and their implementation in a particular object-oriented programming language (e.g. JAVA). This part of the assessment is designed to assess the student's ability to develop complex and concurrent algorithms.

The in-course assignment will assess all Learning Outcomes.

The software artifact described above and assessed formatively as described above will form part of this summative assignment.

Assessment Criteria

The in-course assignment will primarily be marked on the functionality of the code produced, but learners will also be expected to produce useful and well structured documentation.

Learning Strategy

This module requires a two-fold approach to concurrency, algorithms and data structures. A theoretical understanding of the principles, and where necessary modelling techniques, will be delivered by a programme of sessions. This will be reinforced with extensive practical work in the development of algorithms and structures largely to be done as directed unsupervised work. Contact time typically consists of lectures, seminars or studio but is context specific.

Appendix

Indicative references/learning materials:

Algorithms, Data Structures and Complexity

McConnell, J J Analysis of Algorithms: An Interactive Learning Approach
Jones and Bartlett, 2001

Goodrich, M T and Tamassia, R Data Structures and Algorithms in Java
Wiley, 4th Edition, 2005

Watt, D A and Brown, D F Java Collections: An Introduction to Abstract Data Types, Data Structures and Algorithms, Wiley, 2001

Processes

Magee, J and Kramer, J Concurrency: State Models and Java Programs
Wiley, 2nd Edition, 2006