Module Specification

Computational Problem-Solving

Version: 2025-26, v1.0, 20 Jan 2025

Contents

Part 1: Information

Module title: Computational Problem-Solving

Module code: UFCEHQ-15-1

Level: Level 4

For implementation from: 2025-26

UWE credit rating: 15

ECTS credit rating: 7.5

College: College of Arts, Technology and Environment

School: CATE School of Computing and Creative Technologies

Partner institutions: University Centre Weston

Field: Computer Science and Creative Technologies

Module type: Module

Pre-requisites: None

Excluded combinations: None

Co-requisites: None

Continuing professional development: No

Professional, statutory or regulatory body requirements: None

Part 2: Description

Overview: This module introduces students to essential algorithms, data structures, and logic for software development. Covering both theoretical and practical aspects, it prepares students to tackle computational problems effectively, emphasising efficient algorithm design, problem-solving strategies, and the significance of data structure choice on software performance.

Features: Not applicable

Educational aims: •Acquire a deep understanding of key data structures such as arrays, stacks, queues, linked lists, trees, graphs, and hash tables.
•Master fundamental sorting and searching algorithms, understanding their efficiency and application.
•Learn to handle critical sections and race conditions in software development to write more secure and robust code.
•Develop problem-solving skills by applying theoretical knowledge to practical challenges.
•Understand the foundations of algorithms to better predict their performance in real-world applications.

Outline syllabus: •Overview of algorithms, data structures, and their importance in software development.
•Detailed study of both primitive (arrays, vectors, pointers) and abstract data types (linked lists, stacks, queues, trees, hash tables), focusing on their implementation and application
•Exploration of various sorting techniques, including QuickSort, MergeSort, and BubbleSort, with emphasis on their efficiency and use cases.
•Examination of search strategies e.g. Binary Search, Depth-First Search, and Breadth-First Search, understanding their mechanisms and applications.
•Study of asymptotic notation (Big O, Theta, Omega Notation) to analyse algorithm efficiency and performance.
•Fundamentals of algorithm complexity, efficiency analysis, and optimisation techniques to solve complex problems effectively.
•Hands-on exercises and project-based tasks applying data structures and algorithms to real-world challenges.

Part 3: Teaching and learning methods

Teaching and learning methods: Lectures covering the fundamentals of algorithms, data structures and time complexity, followed by practical delivery through a series of workshops, labs, and project-based tasks to develop the skills required to design, implement, and evaluate computational systems. Regular discussions and presentations will foster critical thinking and communication skills.

Module Learning outcomes: On successful completion of this module students will achieve the following learning outcomes.

MO1 Demonstrate the fundamental algorithms, data structures, and their applications in computational problem solving.

MO2 Utilise critical thinking and problem-solving skills to design and implement computational challenges.

MO3 Demonstrate the application of the common data structures used within software development, with their related advantages and disadvantages.

Hours to be allocated: 150

Contact hours:

Independent study/self-guided study = 114 hours

Face-to-face learning = 36 hours

Reading list: The reading list for this module can be accessed at readinglists.uwe.ac.uk via the following link https://rl.talis.com/3/uwe/lists/E4ED1B4E-5589-944E-72FD-36056F317C55.html

Part 4: Assessment

Assessment strategy: The Computational Problem-Solving module is assessed using a 3-hour controlled assessment.

The TCA will assess students' understanding and proficiency in the topics covered in the syllabus. The assessment will evaluate their theoretical and practical knowledge of implementing data structures and algorithms, as well as their ability to critically analyse and solve computational problems.

During the TCA, students will be presented with a specific computational problem. They are required to identify and implement a solution, then write a short evaluation

(Est. 300 words) of their approach, highlighting efficiency, complexity (e.g., using Big O Notation), and possible improvements .

Tutor-led formative feedback will be available throughout the module to support students in their learning and development. Preparation for summative assessment will be supported by opportunity for a mock assessment and formative feedback.

Assessment tasks:

In-class test (First Sit)
Description: 3 hour Time Constrained Assessment
Weighting: 100 %
Final assessment: Yes
Group work: No
Learning outcomes tested: MO1, MO2, MO3

In-class test (Resit)
Description: 3 hour Time Constrained Assessment
Weighting: 100 %
Final assessment: Yes
Group work: No
Learning outcomes tested: MO1, MO2, MO3

Part 5: Contributes towards

This module contributes towards the following programmes of study:

Software Development [UCW] FdSc 2025-26