Prifysgol Wrecsam
Wrexham University

Module specification

When printed this becomes an uncontrolled document. Please access the Module Directory for the most up to date version by clicking on the following link: Module directory

Module Code	COM713
Module Title	Advanced Data Structure and Algorithm
Level	7
Credit value	20
Faculty	FACE
HECoS Code	100956
Cost Code	GACP
Pre-requisite module	N/A

Programmes in which module to be offered

Programme title	Core/Optional/Standalone
MSc Computer Science	Core
MSc Computer Science with Advanced Practice	Core
MSc Cyber Security	Core
MSc Cyber Security with Advanced Practice	Core
MSc Big Data and Data Analytics	Core
MSc Big Data and Data Analytics with Advanced Practice	Core
MSc Artificial Intelligence	Core

Breakdown of module hours

Learning and teaching hours	10 hrs
Placement tutor support hours	0 hrs
Supervised learning hours e.g. practical classes, workshops	11 hrs
Project supervision hours	0 hrs
Active learning and teaching hours total	**21 hrs**
Placement hours	0 hrs
Guided independent study hours	179 hrs
Module duration (Total hours)	**200 hrs**

Module aims

This module aims to give students a thorough grounding in the theories and application of key computer programming concepts such as algorithms, abstract data types, underlying data structures and their integration to produce efficient code. This allows students to develop the knowledge and skills to be able to analyse problems and then design, implement, and analyse effective algorithmic solutions using a suitable programming language. Students will become familiar with the implications of algorithmic solutions in terms of their computational complexity and develop a working knowledge of optimal and approximate solutions to problems. These will be developed using procedural and object-oriented programming with current methodologies to demonstrate proficiency in industry standard techniques.

Module Learning Outcomes

At the end of this module, students will be able to:

1	Analyse and interpret a range of problems and produce designs and models for algorithmic solutions.
2	Critically evaluate problems and solutions in terms of their computational complexity.
3	Articulate and validate the structure of algorithms using computational thinking terminology.
4	Implement computational solutions that demonstrate proficiency in a range of data structures, algorithms and programming techniques.
5	Perform the execution, testing, and debugging of intricate programs, translating the high-level design into tangible programming constructs.

Assessment

Indicative Assessment Tasks:
This section outlines the type of assessment task the student will be expected to complete as part of the module. More details will be made available in the relevant academic year module handbook.

The assignments will be designed to analyse and interpret a range of real-world problems and will require students to identify and interpret a range of problems and produce designs for algorithmic solutions. Students will submit different parts of the portfolio over the semester. The final part of the assessment will allow students to implement computational solutions and demonstrate their skill in writing, compiling, executing, testing and debugging their program.

Overall, the assessments will provide students with the opportunity to apply their skills through solo and team-based programming challenges.

The assessment will require students to compare AI generated code and analyse the differences from their implementation of the program with detailed explanation.

Assessment number	Learning Outcomes to be met	Type of assessment	Duration/Word Count	Weighting (%)	Alternative assessment, if applicable
1	1,2,3,4,5	Portfolio	5000 Words or Equivalent	100%	

Derogations

None

Learning and Teaching Strategies

In line with the Active Learning Framework, this module will be blended digitally with both a VLE and online community. Content will be available for students to access synchronously and asynchronously and may indicatively include first and third-party tutorials and videos, supporting files, online activities any additional content that supports their learning. As this module progresses, the strategies will change to best support a diverse learning environment. Initially, the module will start with a heavier reliance on engaging tutor-led lectures, demonstrations, and workshops to ensure that the students get the relevant threshold concepts. As the module continues experiential and peer learning strategies will be encouraged as the students' progress with their portfolio work. Assessment will occur throughout the module to build student confidence and self-efficacy in relation to their proficiency in a range of data structures, algorithms and programming techniques.

Welsh Elements

This module is designed to support Welsh-speaking students in line with the Welsh Language Standards. While the primary delivery will be in English, students will have the opportunity to submit assessments, including coursework and projects, in Welsh if preferred. Relevant module materials, such as reading lists, key texts, and guidance, will be available bilingually upon request, ensuring accessibility for all students. Additionally, where possible, guest speakers, case studies, or examples may include references to the Welsh business context, especially in areas such as data use in local industries and Welsh public sector organisations.

The department encourages students to develop bilingual digital skills by incorporating Welsh-language datasets, tools, and resources where appropriate, offering an inclusive learning environment. We also support the development of bilingual visualisation techniques, enabling students to create digital outputs that reflect the Welsh language, should they wish to do so.

Indicative Syllabus Outline

- Types of programming languages
- Python programming language
- Algorithms and complexity
- Object-oriented programming
- Stacks, queues and lists
- Recursion
- Searching and sorting
- Tree and graph algorithms

Indicative Bibliography

Please note the essential reads and other indicative reading are subject to annual review and update.

Essential Reads:

F. Romano, B. Baka, & D. Phillips, Getting Started with Python. Packt Publishing, 2019.

Other indicative reading:

P. Barry, P. Head, First Python: A Brain-Friendly Guide. O'Reilly Media, Inc. 2016.

T.H. Cormen, Introduction to Algorithms. 3rd ed. Cambridge, Mass: MIT Press, 2009.

M.T. Goodrich, R. Tamassia, & M.H. Goldwasser, Data structures and algorithms in Python. John Wiley & Sons Ltd. 2013.

Wentworth, P., Elkner, J., Downey, A. B., & Meyers, C. How to Think Like a Computer Scientist. 3rd ed. 2020. Available online: https://buildmedia.readthedocs.org/media/pdf/howtothink/latest/howtothink.pdf

B. Miller, &D. Ranum Problem Solving with Algorithms and Data Structures. Franklin, Beedle & Associates. 2020. Available online: https://runestone.academy/runestone/books/published/pythonds/index.html

Administrative Information

For office use only	
Initial approval date	20/07/2020
With effect from date	Sept 2026
Date and details of revision	Module LOs and assessment update in the Computing Revalidation in July 2023 March 2026 Addition of MSc Artificial Intelligence programme title
Version number	3