PRIFYSGOL Glyndŵr Wrecsam | Wrexham Glyndŵr UNIVERSITY

MODULE SPECIFICATION

When printed this becomes an uncontrolled document. Please access the Module Directory for the most up to date version by clicking here.

Module Code:	COM713

Module Title:	Advanced Data Structures and Algorithms

Level:	7	**Credit Value:**	20

Cost Centre(s):	GACP	JACS3 **code**: HECoS code:	I320 100956

Faculty	FAST	**Module Leader:**	Jessica Muirhead

Scheduled learning and teaching hours	21 hrs
Placement tutor support	0 hrs
Supervised learning eg practical classes, workshops	27 hrs
Project supervision (level 6 projects and dissertation modules only)	0 hrs
Total contact hours	**48** hrs
Placement / work based learning	**0** hrs
Guided independent study	152 hrs
Module duration (total hours)	200 hrs

Programme(s) in which to be offered (not including exit awards)	Core	Option
MSc Computing	✓	□
MSc Computer Science	✓	□
MSc Data Science and Big Data Analytics	✓	□

Pre-requisites
None

Office use only

Initial approval: 22/07/2020 Version no:1

With effect from: 01/09/2020

Date and details of revision: 07/10/2022 AM0 change to number of weekly portfolio tasks from seven to four Version no:

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 (space, time and logical) and develop a working knowledge of optimal and approximate (including heuristic) solutions to problems. These will be developed using object oriented coding and diagramming methodologies to demonstrate proficiency in industry-standard programming techniques.

Module Learning Outcomes - at the end of this module, students will be able to	
1	Demonstrate a critical understanding of programming paradigms
2	Analyse and interpret a range of problems and produce designs and models for algorithmic solutions
3	Identify and evaluate problems and solutions in terms of their computational complexity
4	Explain and justify the structure of algorithms using computational thinking terminology
5	Implement computational solutions that demonstrate proficiency in a range of data structures, algorithms and object-oriented programming techniques
6	Write, compile, execute, test and debug a non-trivial OO program, which maps the high-level design onto concrete programming constructs.

Employability Skills The Wrexham Glyndŵr Graduate	I = included in module content A = included in module assessment N/A = not applicable
CORE ATTRIBUTES	
Engaged	I
Creative	I/A
Enterprising	
Ethical	I
KEY ATTITUDES	
Commitment	
Curiosity	I/A
Resilient	I
Confidence	

Adaptability	I/A
PRACTICAL SKILLSETS	
Digital fluency	I/A
Organisation	A
Leadership and team working	I
Critical thinking	I/A
Emotional intelligence	
Communication	I/A

Derogations

None

Assessment:

Indicative Assessment Tasks:

This module is assessed through a series of four weekly Portfolio tasks designed to test students' understanding of the module content. At the end of the module, a final larger activity will synthesise all of the students' knowledge of data structures and algorithms.

Assessment number	Learning Outcomes to be met	Type of assessment	Weighting (%)
1	1,2,3,4	Portfolio	70%
2	5,6	Project	30%

Learning and Teaching Strategies:

Learning will be delivered through a practical approach, with a series of workshop sessions combining short theory lectures with student-led activities to prepare solutions to simulated problems. In-class sessions will be augmented with guided learning videos and suggested reading to ensure students understand the industry challenges faced by programmers.

Syllabus outline:

1. Types of programming languages
2. Python programming language
3. Algorithms and complexity
4. Object-oriented programming
5. Stacks, queues and lists
6. Recursion
7. Searching and sorting
8. Tree and graph algorithms

Indicative Bibliography:

Essential reading

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

Other indicative reading

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

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

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

Knuth, D.E. (1997) The Art of Computer Programming, Volume 1: Fundamental Algorithms. 3ed. Addison-Wesley.

Miller, B., & Ranum, D. (2013) *Problem Solving with Algorithms and Data Structures.* Franklin, Beedle & Associates. Available online:
https://runestone.academy/runestone/books/published/pythonds/index.html

Neapolitan, R.E. and Naimipour, K. (2014), Foundations of Algorithms. 5th ed. Jones & Bartlett Learning.

Runestone (n.d.) Foundations of Python Programming. Available online:
https://runestone.academy/runestone/books/published/fopp/index.html Sedgewick, R. (2011) Algorithms. 4th ed. Addison-Wesley.

Wentworth, P., Elkner, J., Downey, A. B., & Meyers, C. (2019) How to Think Like a Computer Scientist. 3rd ed. Available online:
https://buildmedia.readthedocs.org/media/pdf/howtothink/latest/howtothink.pdf