MODULE PROFORMA		
Full module title: Software Development II		
Module code:	Credit level: 4	Length: 1 Semester
UK credit value: 20	ECTS value: 10	
College and School: College of Design, Creative and Digital Industries, School of Computer Science and Engineering		
Module Leader(s): Pumudu Fernando / Shamal Perera		
Extension:	Email:	
Host course and course leader: BSc Computer Science		
Status: Core BSc Computer Science, BSc Software Engineering, BSc Data Science and Analytics		
Subject Board: COMENG		
Prerequisites: None	Corequisites: None	
Study abroad: Yes No alternative assessment needed		
Special features: None		
Access restrictions: None		
Are the module learning outcomes delivered, assessed or supported through an arrangement with an organisation(s) other than the University of Westminster: No		
Summary of module content: The module aims to develop skills in the selection and implementation of problemsolving algorithms while learning the Java programming language. It will strengthen abilities in the implementation of algorithms, in terms of adherence to requirements, design and modelling, through to the application of sound programming principles. The understanding of structures and advanced programming methods will also be developed, including sorting, the implementation of classes and methods, as well as more sophisticated data structures such as lists, queues, and stacks.		

Learning outcomes

By the end of the module the successful student will be able to:

LO1 Choose appropriate algorithms and data structures for problem solving and implement these using a programming language;

LO2 Analyse an algorithm to consider its performance in relation to a problem and its data set, and to introduce Big O notation as part of this process.

LO3 Develop solutions to common programming problems using classes to implement fundamental object orientated concepts
LO4 Implement common data structures and data sorting algorithms.
LO5 Undertake basic requirements gathering and data modelling exercises.

Course outcomes the module contributes to:

BSc Computer Science	L4.2, L4.6, L4.7
BEng Software Engineering	L4.3, L4.4, L4.6
BSc Data Science and Analytics	L4.2, L4.4, L4.6

Indicative syllabus content

- Intermediate level algorithms and programming techniques for a range of situations to handle: function calls, recursion, library routines, arrays, exceptions, file handling, and testing and debugging.
- Introduction of advanced programming structures (subroutines and functions) with appropriate data hiding, parameter passing, returning values, local/global variable access, and scope rules.
- Implementation of common data structures using arrays, and other collections (such as stacks, queues, hashtables, trees and graphs), and their use in operations such as data sorting, searching, merging, matching.
- Definition, design and use of classes. Basic object oriented principles.
- Introduction to requirements analysis and data modelling.
- Introduction to programming methodology, the waterfall model. Need for documentation, bug tracking and version control.
- Use of the command line. Understanding class paths, and command line parameter passing.

Teaching and learning methods

Lectures and practical tutorials. The lectures present key theoretical issues, programming exercises, and include practical demonstrations of programming tools and program development. To develop problemsolving skills, students are provided with a set of formative practical programming exercises to be completed both during and outside the scheduled tutorial sessions. Students shall receive feedback on the formative exercises in order to evaluate progress.

Activity type	Category	Student learning and teaching hours*
Lecture/Webcast lecture	Scheduled	22
Tutorial	Scheduled	22

Practical Classes and workshops		12
Total Scheduled		56
Independent study	Independent	144
Total student learning and teaching hours		200

*the hours per activity type are indicative and subject to change.

Assessment rationale

The ability to write reasonably substantial programs following a given design is assessed by means of programming assignments, to be completed by students inside and outside of the tutorial sessions. Formative and summative in class tests are used to assess the students' knowledge of the type of programming techniques and code required for the programming assignments.

Arrays programming assignment 1: LO1, LO2

Object Orientated programming assignment 2: LO3, LO4, LO5

Assessment criteria

Students must demonstrate a proper understanding of the use of the code related with standard algorithms and coding for the programming assignment assessment. To achieve higher marks, students must demonstrate the design and use of algorithms, techniques and code instructions of increased sophistication, that have been developed independently to produce a program of increased complexity.

Assessment methods and weightings

Assessment name	Weighting %	Qualifying mark %	Qualifying set	Assessment type (e.g. essay, presentation, open exam or closed exam)
In class test	*50*	*30*		*in-class test component*
Coursework	*50*	*30*		*Coding assignment*

Synoptic assessment

No synoptic assessment.

Sources

Course Notes and reading lists on the Blackboard, up to date WWW references.

Link to the online reading list

https://rl.talis.com/3/westminster/lists/80D25FB9-0B7E-AEE3-9873-92150E0D0B9A.html