UNIVERSITY OF LONDON

BSc Computer Science

Module Specification

Key Information			
Module title	Object Oriented Programming		
Level	5	Credit value	15
Member Institution	Goldsmiths	Notional study hours and duration of course	150
Module lead author/ Subject matter expert			
Module co-author			

Rationale for the module

There are several programming paradigms in computer science. One of the most important, especially when creating large software projects, is object-oriented programming, or OOP. OOP allows the integration of data and processes on that data into discrete software modules. Learning about OOP will allow you to develop more modularised, more complex software designs. It will also let you understand better how many existing software libraries and systems have been designed. In the later stages of the programme, OOP is used extensively.

Aims of the module

This module aims to provide you with an object-oriented programming skill set. You will learn what objects and classes are and how to write your classes. You will see how objects can interact with each other, including defining and implementing interfaces to control the interaction. You will learn how to use inheritance to inherit and extend functionality from parent classes. You will learn how to write code according to style guidelines and how to write formal code documentation.

Topics covered in this module:

The topics listed here are an approximation of what will be covered. The topics presented may be slightly revised to ensure currency and relevance. Students will be advised of any changes in advance of their study.

1. Text I/O and functions
2. Using classes and variables to model data
3. File I/O, exception handling and algorithms
4. Writing and testing an algorithm
5. Object interactions
6. Libraries, toolkits, frameworks and widgets
7. Event driven programming and inheritance
8. Refactoring and class design
9. Initialiser lists, constructors and threads
10. Advanced class and user interfaces

Approximately 10-12 hours of study will be required per topic. The remaining study time is intended for coursework.

Learning outcomes for the module

Students who successfully complete this module will be able to:

1. Understand and explain the key principles of object oriented programming
2. Choose appropriate basic data types to represent different data
3. Write classes with data and functions
4. Explain the purpose of interfaces and write classes that implement specific interfaces
5. Use inheritance to implement a hierarchy of classes
6. Write formal code documentation and write code following style guidelines

Assessment strategy, assessment methods

Summative and Formative Assessments
The module will contain a range of summative and formative assessments. Summative assessments are assessments which contribute directly towards your final grade. Formative assessments do not count directly towards your final grade. Instead, they provide you with opportunities for low stakes practice, and will often provide some sort of feedback about your progress. For example, a practice quiz might provide you with feedback about why a particular answer was wrong.

Assessment Activities
The table below lists the assessment activity types you might encounter taking the module. It also states if that type of assessment can be automatically graded. For example, multiple choice quizzes can be automatically graded, and so can some programming assignments. It also states if that type of assessment will be found in the summative coursework. More details about the summative assessments are provided below.

Assessment activity type	Can it be automatically graded with feedback in some cases?	Coursework
Quiz	X	X
Writing task		X
Programming task	X	X
Peer review task		x

Pass Mark

In order to pass this module, you must achieve at least 35% in each element of summative assessment and an overall weighted average of 40%, subject to the application of rules for compensation. Please refer to the programme regulations for more information.

Summative Assessment Elements

This is a module that is best assessed largely through continuous assessment by way of programming exercises worked on throughout the term.

Summative Assessment Component	Components	Percentage of final credit	Deadline
Coursework 1	Four programming exercise submissions	50%	Mid session
Coursework 2	Four programming exercise submissions	50%	End of session

Each of the two courseworks will take up to 25 hours of study time to complete and comprise a variety of practical exercises and quizzes.

Learning resources

The module will draw on a number of different, largely web-based, public resources as well as the resources produced as bespoke material for this module.

The programming language used will be C++.

Specific essential readings for this module will be taken from the following text book:

> Ivor Horton and Peter Van Weert , Beginning C++17: From Novice to Professional, Apress, 2018, Fifth edition ISBN-13 (pbk): 978-1-4842-3365-8