



MODULE SPECIFICATION

Part 1: Information			
Module Title	C++ Development		
Module Code	UFCFBF-15-2	Level	Level 5
For implementation from	2019-20		
UWE Credit Rating	15	ECTS Credit Rating	7.5
Faculty	Faculty of Environment & Technology	Field	Computer Science and Creative Technologies
Department	FET Dept of Computer Sci & Creative Tech		
Module type:	Standard		
Pre-requisites	Programming in C 2019-20		
Excluded Combinations	None		
Co- requisites	None		
Module Entry requirements	None		

Part 2: Description
<p>Overview: Pre-requisites: students must take one out of UFCFWA-30-1 Entertainment Software Development or UFCFF6-30-1 Programming in C</p> <p>Educational Aims: See Learning Outcomes</p> <p>Outline Syllabus: Rationale for using C++ in Software Development</p> <p>C++ language features:</p> <ul style="list-style-type: none"> Memory allocation / deallocation Object orientation: inheritance and polymorphism Exception handling Templates Operator overloading

STUDENT AND ACADEMIC SERVICES

Delegate functions

Compiler directives

Unmanaged code:

Automatic vs dynamic memory handling

Standard Template Library

Measuring and analysing performance

Memory alignment, bit manipulation, packing, pooling

Teaching and Learning Methods: Contact time: 36 hours

Assimilation and development of knowledge: 74 hours

Exam preparation: 10 hours

Coursework preparation: 30 hours

Total study time: 150 hours

Lectures will be used to introduce relevant programming concepts whilst being practically explored within supervised studio sessions guided by tutorial tasks.

A set number of the tutorial tasks are to be completed to form individual lab logbooks.

Aside from the tutorial tasks, students will be set a small number of more challenging tasks to implement taught concepts, using supplied designs / code / libraries / SDKs where appropriate. It is expected that the majority of this work will be carried out independently, outside of taught sessions, though specific sessions will be organised to provide targeted help with these tasks prior to hand-in.

Part 3: Assessment

Formative assessment:

The tutorial tasks set for the module will be peer and tutor reviewed regularly in studio/practical sessions. Completed tasks will contribute to a logbook, which forms part of the students' portfolios. While this logbook contributes to the summative assessment, it is assessed on a pass/fail basis only, and is designed to encourage student engagement.

Summative assessment:

In addition to the tutorial tasks, a small number of more challenging tasks will be set. These tasks form the summative part of the portfolio for the module, and will be set in order of increasing complexity/weighting. The reason behind this strategy is to align assessed tasks with the topics being taught, and distribute workload for the module across the year.

A final assessment for the module will ensure detailed understanding of language mechanisms that form part of several learning outcomes but cannot easily be assessed through practical tasks.

STUDENT AND ACADEMIC SERVICES

First Sit Components	Final Assessment	Element weighting	Description
Portfolio - Component B		75 %	Portfolio of practical exercises and lab logbook
Presentation - Component A	✓	25 %	Presentation / demonstration
Resit Components	Final Assessment	Element weighting	Description
Portfolio - Component B		75 %	Portfolio of practical exercises
Presentation - Component A	✓	25 %	Presentation / demonstration

Part 4: Teaching and Learning Methods															
Learning Outcomes	<p>On successful completion of this module students will achieve the following learning outcomes:</p> <table border="1"> <thead> <tr> <th>Module Learning Outcomes</th> <th>Reference</th> </tr> </thead> <tbody> <tr> <td>Analyse the impact of using various C++ language features on the compilation process for non-trivial software projects</td> <td>MO1</td> </tr> <tr> <td>Demonstrate an in-depth understanding of the run-time behaviour of a C++ application, and the significance of the call-stack</td> <td>MO2</td> </tr> <tr> <td>Design and implement object orientated applications that make appropriate use of mechanisms such as polymorphism, templates and delegate functions</td> <td>MO3</td> </tr> <tr> <td>Apply their understanding of issues surrounding memory management within C++, to develop object oriented applications which avoid issues such as memory leaks, pointer errors and undefined behaviour</td> <td>MO4</td> </tr> <tr> <td>Recognise issues related to efficiency and organisation of memory resources within unmanaged code and apply strategies to reduce their impact on run-time performance</td> <td>MO5</td> </tr> <tr> <td>Discuss the role and significance of external libraries and Software Development Kits (SDKs), their relationship to C++ and their role in crossplatform development</td> <td>MO6</td> </tr> </tbody> </table>	Module Learning Outcomes	Reference	Analyse the impact of using various C++ language features on the compilation process for non-trivial software projects	MO1	Demonstrate an in-depth understanding of the run-time behaviour of a C++ application, and the significance of the call-stack	MO2	Design and implement object orientated applications that make appropriate use of mechanisms such as polymorphism, templates and delegate functions	MO3	Apply their understanding of issues surrounding memory management within C++, to develop object oriented applications which avoid issues such as memory leaks, pointer errors and undefined behaviour	MO4	Recognise issues related to efficiency and organisation of memory resources within unmanaged code and apply strategies to reduce their impact on run-time performance	MO5	Discuss the role and significance of external libraries and Software Development Kits (SDKs), their relationship to C++ and their role in crossplatform development	MO6
Module Learning Outcomes	Reference														
Analyse the impact of using various C++ language features on the compilation process for non-trivial software projects	MO1														
Demonstrate an in-depth understanding of the run-time behaviour of a C++ application, and the significance of the call-stack	MO2														
Design and implement object orientated applications that make appropriate use of mechanisms such as polymorphism, templates and delegate functions	MO3														
Apply their understanding of issues surrounding memory management within C++, to develop object oriented applications which avoid issues such as memory leaks, pointer errors and undefined behaviour	MO4														
Recognise issues related to efficiency and organisation of memory resources within unmanaged code and apply strategies to reduce their impact on run-time performance	MO5														
Discuss the role and significance of external libraries and Software Development Kits (SDKs), their relationship to C++ and their role in crossplatform development	MO6														
Contact Hours	<table border="1"> <thead> <tr> <th colspan="2">Independent Study Hours:</th> </tr> </thead> <tbody> <tr> <td>Independent study/self-guided study</td> <td>114</td> </tr> <tr> <td>Total Independent Study Hours:</td> <td>114</td> </tr> <tr> <th colspan="2">Scheduled Learning and Teaching Hours:</th> </tr> <tr> <td>Face-to-face learning</td> <td>36</td> </tr> <tr> <td>Total Scheduled Learning and Teaching Hours:</td> <td>36</td> </tr> <tr> <td>Hours to be allocated</td> <td>150</td> </tr> </tbody> </table>	Independent Study Hours:		Independent study/self-guided study	114	Total Independent Study Hours:	114	Scheduled Learning and Teaching Hours:		Face-to-face learning	36	Total Scheduled Learning and Teaching Hours:	36	Hours to be allocated	150
Independent Study Hours:															
Independent study/self-guided study	114														
Total Independent Study Hours:	114														
Scheduled Learning and Teaching Hours:															
Face-to-face learning	36														
Total Scheduled Learning and Teaching Hours:	36														
Hours to be allocated	150														

STUDENT AND ACADEMIC SERVICES

	Allocated Hours	150
Reading List	<p><i>The reading list for this module can be accessed via the following link:</i></p> <p>https://uwe.rl.talis.com/modules/ufcfbf-15-2.html</p>	

Part 5: Contributes Towards

This module contributes towards the following programmes of study:

Electronic and Computer Engineering [Sep][FT][Frenchay][3yrs] BEng (Hons) 2018-19

Electronic and Computer Engineering [Sep][SW][Frenchay][4yrs] BEng (Hons) 2018-19

Electronic and Computer Engineering {Top Up} [Aug][FT][SHAPE][1yr] BEng (Hons) 2018-19

Electronic and Computer Engineering {Top Up} [Aug][PT][SHAPE][2yrs] BEng (Hons) 2018-19

Electronic and Computer Engineering [Sep][PT][GlosColl][5yrs] BEng (Hons) 2018-19

Electronic and Computer Engineering {Apprenticeship} [Sep][PT][GlosColl][5yrs] BEng (Hons) 2018-19