MODULE SPECIFICATION

Part 1: Information			
Module Title	Entertainment Software Development		
Module Code	UFCFWA-30-1	Level	Level 4
For implementation from	2018-19		
UWE Credit Rating	30	ECTS Credit Rating	15
Faculty	Faculty of Environment & Technology	Field	Computer Science and Creative Technologies
Department	FET Dept of Computer Sci & Creative Tech		
Contributes towards	Games Technology [Sep][SW][Frenchay][4yrs] BSc (Hons) 2018-19 Digital Media [Sep][FT][Frenchay][3yrs] BSc (Hons) 2018-19 Games Technology [Sep][FT][Frenchay][3yrs] BSc (Hons) 2018-19 Digital Media [Sep][FT][Frenchay][3yrs] BSc (Hons) 2018-19 Digital Media [Sep][SW][Frenchay][4yrs] BSc (Hons) 2018-19 Digital Media [Sep][SW][Frenchay][4yrs] BSc (Hons) 2018-19 Digital Media [Sep][FT][SHAPE][3yrs] BSc (Hons) 2018-19 Digital Media [Sep][FT][SHAPE][3yrs] BSc (Hons) 2018-19		
Module type:	Standard		
Pre-requisites	None		
Excluded Combinations	None		
Co- requisites	None		
Module Entry requirements	None		

Part 2: Description

Educational Aims: The aim of this module is to introduce students to fundamental concepts underpinning computer games programming, including the C++ programming language and fundamental software development practices, problem solving techniques and mathematics which, together, will allow students to confidently write code that solves typical games programming problems.

Outline Syllabus: Below is a list of module topics.

Introduction to the C++ programming language:

Variables and operators
Control structures and execution flow
Functions
Classes and object orientation
IDEs and compilation / execution / debugging processes

Software development process:

Problem solving with code / functional decomposition, from planning to implementation
Object oriented decomposition and UML notation
Testing strategies
Coding style considerations and documentation practices
Hardware resource implications and routes for optimisation
An introduction to threading and related software design implications

Mathematics:

Set theory and logic: operators, truth tables, simple propositional / predicate logic.
Computer arithmetic: binary, decimal and hexadecimal representations.
Algebra: basic manipulation, Cartesian coordinates, lines, curves and linear equations.
Trigonometry, functions, tangents, and normals as applied to geometry.

Teaching and Learning Methods: This module will involve 6 hours contact time per fortnight. The time will be divided between lectures and studio sessions as appropriate. Extra, targeted, drop in sessions may be arranged prior to portfolio hand-ins.

Contact time: 72 hours
Assimilation and development of knowledge: 148 hours
Exam preparation: 20 hours
Coursework preparation: 60 hours
Total study time: 300 hours

Lectures will introduce 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 where appropriate. It is expected that the majority of this work will be carried out independently, outside of taught sessions, though assessment 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. These will be assessed through inclass demos.

A final examination for the module will assess detailed understanding of taught material that form part of several learning outcomes but cannot easily be assessed through practical tasks.

First Sit Components	**Final Assessment**	**Element weighting**	**Description**
Portfolio - Component B		75 %	Portfolio of practical exercises and lab logbook
Examination - Component A	✓	25 %	Examination (2 hours
Resit Components	**Final Assessment**	**Element weighting**	**Description**
Portfolio - Component B		75 %	Portfolio of practical exercises and lab logbook
Examination - Component A	✓	25 %	Examination

Part 4: Teaching and Learning Methods

Learning Outcomes

On successful completion of this module students will be able to:

	Module Learning Outcomes
MO1	Write, compile and run high-level computer programs demonstrating appropriate use of the C++ language syntax
MO2	Utilise the debugging facilities of an IDE (such as Visual Studio) to identify, analyse and resolve run-time errors
MO3	Use methodical processes to analyse and decompose typical games programming problems in order to design, implement and evaluate their algorithmic solutions
MO4	Employ software engineering techniques and associated notation to illustrate and interpret small-scale software designs
MO5	Apply fundamental mathematical concepts from algebra, trigonometry, computational arithmetic, logic and set theory, to solve games programming problems
MO6	Discuss the role of threading in computer programming and its impact on program design

Contact Hours

Contact Hours	
Independent Study Hours:	
Independent study/self-guided study	228
Total Independent Study Hours:	228
Scheduled Learning and Teaching Hours:	
Face-to-face learning	72
Total Scheduled Learning and Teaching Hours:	72
Hours to be allocated	300
Allocated Hours	300

Reading List

The reading list for this module can be accessed via the following link:

https://uwe.rl.talis.com/modules/ufcfwa-30-1.html