MODULE SPECIFICATION

Part 1: Information			
Module Title	Programming in C		
Module Code	UFCFF6-30-1	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	Computing [Sep][SW][Frenchay][4yrs] BSc (Hons) 2018-19 Computing {Dual} [Mar][SW][Taylors][4yrs] BSc (Hons) 2018-19 Computing {Dual} [Aug][SW][Taylors][4yrs] BSc (Hons) 2018-19 Electronic and Computer Engineering [Sep][FT][Frenchay][3yrs] BEng (Hons) 2018-19 Computing [Sep][FT][Frenchay][3yrs] BSc (Hons) 2018-19 Computing {Dual} [Mar][FT][Taylors][3yrs] BSc (Hons) 2018-19 Computing {Dual} [Aug][FT][Taylors][3yrs] BSc (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		
Module type:	Standard		
Pre-requisites	None		
Excluded Combinations	None		
Co- requisites	None		
Module Entry requirements	None		

Part 2: Description

Educational Aims: See Learning Outcomes

Outline Syllabus: Basic syntax of ISO90 C to support a structured approach to program development using procedural abstractions - program and control structures - basic data types - reuse of basic functions for I/O, string and mathematical manipulation.

Structured types - arrays, vectors and classes as records. Algorithm design. Simple file processing.

The use of functions and parameters. Global and local variables.

Bit-wise and logical operators.

Using pointers for accessing data and evoking functions.

Problem analysis and design using a structured, step-wise refinement approach.

Structure charts as effective documentary aids for HLL programs.

Introduction to the use of FSDs for event-driven applications, and the implementation of FSMs as FSTs.

Elementary introduction to the problems of multi-tasking.

Speed control for a small DC motor. Driving a stepper motor from a processor.

Use of make files and project configuration specs.

An introduction to data structures.

An introduction to object oriented concepts including classes, objects, inheritance and polymorphism.

Teaching and Learning Methods: This module will involve 6 hours contact time per fortnight. The time will be more or less equally divided between lecture sessions, laboratory sessions.

Contact time: 72 hours
Assimilation and development of knowledge: 148 hours
Exam preparation: 40 hours
Coursework preparation: 40 hours
Total study time: 300 hours

The module will be taught with a very strong emphasis on practical work and the development of understanding by numerous demonstrations and simple, progressive exercises.

The first half of the course will concentrate on teaching basic syntax and use of a structured, stepwise-refinement approach to design and implementation with exposure to structure charts and finite state diagrams.

The second half of the course will develop an understanding of the difficulties involved with I/O programming. Event driven programs will be implemented using finite state methods.

An extended case-study, supported by focussed laboratory based workshops, will allow the students to follow through an example application from design to implementation, and appreciate the relevance of all the component parts of the module syllabus.

Lectures will be used to introduce concepts, syntax and design methods. Laboratory sessions (workshops) will be used to practice and reinforce the students understanding of these. Students

will be expected to work for an equivalent amount of their own time independently on the workshop material, and to independently read their reference book.

The module will be supported by the Faculty's Peer Assisted Learning (PAL) programme. Please see the Faculty web pages for more details of the programme.

Scheduled learning includes lectures and workshops.
Independent learning includes hours engaged with essential reading, assignment preparation and completion etc.

Part 3: Assessment

Assessment will be by two practical exercises and a formal examination. This strategy has been chosen as the examination tests the student's knowledge of the theory that they require in order to be successful at the coursework, thus reducing the impact of issues such as collusion.

The first coursework exercise will contribute only a small amount to the overall assessment. This will enable students to receive written feedback on their progress. In class exercises will be used formatively and will be another opportunity for students to receive feedback. A significant percentage of the marks will be awarded for the students demonstrating and explaining their work.

The second coursework exercise will be a substantial control based, problem solving programming exercise. Students may choose to complete this individually or in pairs. This will contribute the bulk of the component B assessment. The overwhelming majority of the marks for this work will be awarded through the student demonstrating and explaining their work.

A formal examination will be used to enable the students to demonstrate their understanding of C programming and basic design.
This examination will be set as a two hour paper.

First Sit Components	**Final Assessment**	**Element weighting**	**Description**
Set Exercise - Component B		10 %	Practical coursework 1 involving the development of programme code.
Set Exercise - Component B		40 %	Practical coursework 2 involving the development and demonstration of programme code.
Examination - Component A	✓	50 %	Examination (2 hours)
Resit Components	**Final Assessment**	**Element weighting**	**Description**
Set Exercise - Component B		50 %	Practical exercises involving the development of programme code.
Examination - Component A	✓	50 %	Examination (2 hours

Part 4: Teaching and Learning Methods

Learning Outcomes

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

	Module Learning Outcomes
MO1	Show a detailed knowledge of the C programming language
MO2	Demonstrate an understanding of finite state design
MO3	Demonstrate an understanding of structural design approaches
MO4	Demonstrate problem solving and programming skills
MO5	Understand the structures involved in both procedural and object oriented languages

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/ufcff6-30-1.html