Module Specification

Programming in C

Version: 2023-24, v2.0, 17 Mar 2023

Contents

Part 1: Information

Module title: Programming in C

Module code: UFCFF6-30-1

Level: Level 4

For implementation from: 2023-24

UWE credit rating: 30

ECTS credit rating: 15

Faculty: Faculty of Environment & Technology

Department: FET Dept of Computer Sci & Creative Tech

Partner institutions: None

Delivery locations: Not in use for Modules

Field: Computer Science and Creative Technologies

Module type: Module

Pre-requisites: None

Excluded combinations: None

Co-requisites: None

Continuing professional development: No

Professional, statutory or regulatory body requirements: None

Part 2: Description

Overview: Not applicable

Features: Not applicable

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.

Part 3: Teaching and learning methods

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.

Module Learning outcomes: On successful completion of this module students will achieve the following 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

Hours to be allocated: 300

Contact hours:

Independent study/self-guided study = 228 hours

Face-to-face learning = 72 hours

Total = 300

Reading list: The reading list for this module can be accessed at readinglists.uwe.ac.uk via the following link https://uwe.rl.talis.com/modules/ufcff6-30-1.html

Part 4: Assessment

Assessment strategy: Assessment will be by 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 coursework will be in the form of a portfolio of work with several practical exercises to be completed throughout the module run. There will be multiple opportunities for formative feedback. All associated learning outcomes will be assessed. A significant percentage of the marks will be awarded for the students 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.

Assessment components:

Set Exercise (First Sit)
Description: Practical exercises involving the development of programme code.
Weighting: 50 %
Final assessment: No
Group work: No
Learning outcomes tested: MO4

Examination (First Sit)
Description: Examination (2 hours)
Weighting: 50 %
Final assessment: Yes
Group work: No
Learning outcomes tested: MO1, MO2, MO3, MO5

Set Exercise (Resit)
Description: Practical exercises involving the development of programme code.
Weighting: 50 %
Final assessment: No
Group work: No
Learning outcomes tested:

Examination (Resit)

Description: Examination (2 hours)

Weighting: 50 %

Final assessment: Yes

Group work: No

Learning outcomes tested:

Part 5: Contributes towards

This module contributes towards the following programmes of study: