# Module Specification

## Programming for Cyber Security

Version: 2026-27, v2.0, Approved

**Contents**

## Part 1: Information

**Module title:** Programming for Cyber Security

**Module code:** UFCFGL-30-1

**Level:** Level 4

**For implementation from:** 2026-27

**UWE credit rating:** 30

**ECTS credit rating:** 15

**College:** College of Arts, Technology and Environment

**School:** CATE School of Computing and Creative Technologies

**Partner institutions:** None

**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:** This module covers fundamental programming concepts, such as variables, data types, control structures, functions, arrays, I/O operations, secure coding practices and programming paradigms. Students will also learn how to switch from one to another programming language rapidly. Emphasis will be given on developing problem-solving skills, algorithmic and logical thinking, and good coding practices.

**Features:** Not applicable

**Educational aims:** This module aims to give students a solid foundational knowledge and understanding of how programming languages are used to develop solutions for variety of problems and scenarios through a practice-led approach. The module also aims to prepare students for completing challenging coding activities in their degree ahead.

**Outline syllabus:** Topics are likely to include but are not limited to:

Introduction to logical thinking and problem solving
Introduction to C and Linux
Variables and data types
Iteration (for, while, do-while)
Selection (if, else if, switch)
Arrays, Functions , Pointers, Strings and Files handling
Introduction to memory management
Introduction to programming paradigms: Switch from C to Python
Lists, tuples, dictionaries, and sets
File handling operations
Secure programming (basics and advance)
Advanced programming concepts

## Part 3: Teaching and learning methods

**Teaching and learning methods:** Teaching and learning will be based on a triad of theory (concept), analysis (logical thinking and problem solving), and practice (implementation and experimentation). Moreover, the teaching will be carried out through series of bitesize videos and interactive quiz sessions, and the learning will be achieved during the practical sessions.

Students will be rewarded with attractive digital badges upon completion of each major milestone such as the implementation of a challenging scenario-based application based on the covered topics. This practice will continue till end of the module.

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

**MO1** Demonstrate and apply knowledge of essential programming language concepts and analyse their application in solving technical problems.

**MO2** Design, develop and implement algorithmic solutions to programming problems, demonstrating systematic problem-solving and code development skills.

**MO3** Evaluate security vulnerabilities in code and apply defensive programming techniques to develop secure applications.

**Hours to be allocated:** 300

**Contact hours:**

Independent study/self-guided study = 228 hours

Face-to-face learning = 72 hours

**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/ufcfgl-30-1.html

## Part 4: Assessment

**Assessment strategy:** Summative assessment for both the first sit and resit will comprise an in-class test and a portfolio of practical tasks. These assessments are designed to evaluate students' logical reasoning, problem-solving ability, and practical competence. The in-class test will address foundational concepts whereas the portfolio will address both foundational and advanced concepts introduced throughout the module, with particular emphasis on code development that underpins programming skills relevant to cyber security.

Formative assessment will support students during the first sit through structured activities, including mock tests and regular practical exercises such as case studies. These activities aim to reinforce understanding and provide opportunities to

implement a range of programming solutions in preparation for the summative assessment.

The resit assessment will follow the same format as the first sit. Tasks might be scaled appropriately.

**Assessment tasks:**

**In-class test** (First Sit)
Description: Practical in-class test
Weighting: 50 %
Final assessment: No
Group work: No
Learning outcomes tested: MO1, MO3

**Portfolio** (First Sit)
Description: A portfolio of up to 3 practical tasks.
Weighting: 50 %
Final assessment: Yes
Group work: No
Learning outcomes tested: MO2, MO3

**In-class test** (Resit)
Description: Practical in-class test
Weighting: 50 %
Final assessment: No
Group work: No
Learning outcomes tested: MO1, MO3

**Portfolio** (Resit)
Description: A portfolio of up to 3 practical tasks.
Weighting: 50 %
Final assessment: Yes
Group work: No

Learning outcomes tested: MO2, MO3

## Part 5: Contributes towards

This module contributes towards the following programmes of study:

Cyber Security and Digital Forensics {Foundation} [Frenchay] BSc (Hons) 2025-26

Cyber Security and Digital Forensics [Frenchay] BSc (Hons) 2026-27

Cyber Security and Digital Forensics [Frenchay] BSc (Hons) 2026-27

Cyber Security and Digital Forensics {Foundation} [Frenchay] BSc (Hons) 2025-26

Cyber Security and Digital Forensics [NepalBrit] BSc (Hons) 2026-27