Sheffield Hallam University

MODULE DESCRIPTOR

TITLE	FUNDAMENTALS OF PROGRAMMING FOR SOFTWARE ENGINEERING
MODULE CODE	55-407291
LEVEL	4
CREDITS	20
FACULTY	Science Technology And Arts
DEPARTMENT	Computing - STA
SUBJECT GROUP	SOFTWARE ENGINEERING AND COMPUTER SCIENCE - STA
COLLABORATIVE PARTNER / LOCATION (If applicable)	

TOTAL NUMBER OF NOTIONAL STUDY HOURS FOR THIS MODULE **Based on 10 notional study hours per credit**	200
TOTAL NUMBER OF SCHEDULED LEARNING AND TEACHING ACTIVITIES	49
TOTAL NUMBER OF INDEPENDENT LEARNING HOURS **Including time allowed for assessment activities**	152
TYPICAL NUMBER OF SCHEDULE LEARNING AND TEACHING ACTIVITIES PER WEEK	2

MODULE LEARNING OUTCOMES

Describe, recognise and deploy key concepts that relate to designing small imperative applications
Describe, recognise and deploy essential features of a mainstream programming language (such as C/C++) and use them to implement solutions to a variety of programming problems, selecting appropriate control constructs and data structures
Select and apply appropriate software tools and program testing techniques on small programs

MODULE SUMMARY (includes indicative content)

This module provides an introduction to computer programming. It will provide a foundation and understanding of key programming concepts and their deployment in a mainstream imperative programming language. The focus is on the design and implementation of programs using the imperative facilities of a programming language, which is widely used for developing modern applications.

The module offers an opportunity to acquire practical skills that are deemed fundamental to the computing subject area and constitutes a levelling experience that acknowledges the fact that learners entering the programme possess a wide variety of programming skills and experience.

By the end of the module, students should be able to construct simple applications using an imperative programming language from only basic descriptions using a combination of programming experience and problem-solving.

Indicative content:

- **General Programming Concepts** - Problem solving, top-down design and functional decomposition • the development process - specification, design, implementation, testing
- **Variables and Data Structures** - Data types - character, integer, floating point, boolean, arrays • assignment • input/output • string • file handling • objects
- **Algorithms and Control Structures** - Algorithmic design • control structures - sequence, selection, repetition • functions, parameters, return values
- **Program Quality** - Production of correct, understandable and maintainable code • documentation and readability • testing
- **Software Tools and Program Development Environments** - Compilers • debuggers • libraries • version control

LEARNING, TEACHING AND ASSESSMENT SUMMARY

Students will be supported in their learning, to achieve the learning outcomes, in the following ways:

- Each week students will have two hours during which they will be shown concepts that they will then apply to solve problems.

- Elements of lecture-type delivery will be used when appropriate, although the overall emphasis will be a practical-based approach in keeping with the nature of the topics covered in this module; students will be given the opportunity to discuss and apply concepts and principles in tutorials and laboratory work which will have a practical problem solving nature.

- Students will have the opportunity to review their progress on the module via a number of non-assessed test questions, with the opportunity to discuss these questions during tutorial time.

- The module will be supported by the Blackboard on-line learning environment. All teaching materials, questions, additional background reading and supporting materials will be posted here. The Blackboard site will also be used to support the module assessment.

Feedback

Students will receive feedback on their performance in the following ways

- In class, verbally, each week as part of ongoing tutorials and lab sessions

- Detailed written (or audio/video) feedback for coursework elements will be provided using marking grids were appropriate. Additional feedback and clarification can be obtained during tutorial sessions.

- Blackboard will be used to provide students with performance grades and assessment feedback, as well as additional information that may be requested as a result of student feedback throughout the progress of the module

Assessment

The assessment for this module will focus on the student's practical programming skills. Students will demonstrate their knowledge and understanding of the taught principles in this module using programming tasks selected for their relevance to the course using coursework and exam. Assessment will include live-coding and programming exercises across a mix of individual and group tasks subject to the complexity of the programming challenges.

ASSESSMENT INFORMATION

Task No.	Assessment Task Description (e.g. essay, artwork, journal etc)	Word Count or Exam Duration	Task Weighting %	Assessment Task Type Coursework (CW) Written Exam (EX) Practical (PR)
001	Written Assignment		75	CW
002	Written Examination	03:00	25	EX

LEARNING RESOURCES FOR THIS MODULE

https://shu.rl.talis.com/modules/55-407291.html (https://shu.rl.talis.com/modules/55-407291.html)