**MODULE DESCRIPTOR**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Module Title** | **Introduction to Programming** | | | | | |
| **Module Code** | CO1404 (L4) | **JACS CODE** | | I100 | **CREDIT**  **VALUE** | 10 |
| **date OF**  **ApprovAL** | April 2017 | | | | **VERSION NUMBER** | 6 |
| **SCHOOL** | Physical Sciences and Computing | | **PARTNER INSTITUTION** | | **UCL Sri Lanka** | |

**Relationship with other Modules**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Co-requisites** | None | **Pre-requisites** | NONE | **Excluded Combinations** | NONE |

**Module Aims**

|  |
| --- |
| 1. To develop the students’ ability to analyse a simple problem with a view to implementing a software solution. 2. To provide students’ with core skills in computer programming and debugging necessary to implement a readable and maintainable solution to a given problem. |

**MODULE Content**

|  |
| --- |
| In this module students learn how to write simple programs. No prior programming experience is assumed. To provide context and motivation, examples and practical exercises relate loosely to the different courses that the students may progress onto. This module is specified as using C#, but could be delivered using a different programming language.  The module uses a single programming language and programming environment, but develops transferable programming skills by focusing on the fundamental concepts used in the majority of programming languages. From the outset, attention will be paid to program readability and maintainability.  Implementation  Introduction to C# (or other language)  Overall program structure and program flow  Variables and data types  Assignment and simple operators  Simple I/O  Conditions (if statements)  Iteration (for and while loops)  Functions  Simple containers (e.g. array or list depending on language)  Readability and maintainability of program code (e.g. layout, comments, constants)  Use of a Programming Environment  Editor, compiler, debugger and on-line help  Program Testing and Debugging |

|  |  |
| --- | --- |
| **Learning Outcomes** | |
| **On successful completion of this module a student will be able to:** | |
| **1.** | Develop a structured solution to a simple problem. |
| **2.** | Explain the importance of code readability and maintainability. |
| **3.** | Check the robustness of the code using an appropriate test strategy. |

**ASSESSMENT METHODS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| The method of assessment for this module has been designed to test all the learning outcomes. Students must demonstrate successful achievement of these learning outcomes to pass the module. | | | | | |
| **Number of Assessments** | **Form of Assessment** | **% weighting** | **Size of Assessment/Duration/**  **Wordcount** (indicative only) | **Category of assessment**  **(**1 of written exam/practical assessment/coursework –see guidance notes) | **Learning Outcomes being assessed** |
| 1 | Programming assignment | 100% | Practical development equivalent to 2,000 words | Coursework | 1,2,3 |

**Module Pass Requirements**

|  |
| --- |
| To pass this module you must achieve a mark of 40% or above. |

**appendix**

**MODULE CODE: CO1404**

**MODULE TITLE: Introduction TO Programming**

**location of study:** UCLan campus, UCL Sri Lanka

|  |  |
| --- | --- |
| **Module TUTOR(S)** | Dr Daniel George Campbell |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Module**  **Delivery** | Semester Long | Semester 1 | **✓** | Semester 2 | | |  | Semester 3 | |  |
| Year long | Semester 1 & 2 | | |  | Semester 2 & 3 | | |  | |
| Other (please indicate pattern of delivery) | May also be delivered in burst mode (1 week) | | | | | | | | |

**Module Learning Plan**

|  |  |
| --- | --- |
| **Learning, teaching AND ASSESSMENT Strategy**  Lectures introduce the fundamental concepts of programming, cross-referenced to recommended texts.  Practical sessions present a range of simple problems that require the use of the concepts introduced in the lectures. Exercises each week begin simple (e.g. adjust the functionality of a given, working program) and increase in difficulty until the student is writing small programs from scratch. Advanced exercises are provided to stretch those students who adapt quickly to the module.  Support sessions provide extra guidance to help students progress with the practical material, and also to enthuse those who are interested in taking the exercises further. In addition, there will be guided workshops on the use of a programming environment and debugging programs.  Assessment consists of:  A practical assignment, which will reinforce and extend the work covered in practical classes.  An exam to assess the student’s understanding of the theory presented in the module. | |
|  | |
| **SCHEDULED LEARNING AND TEACHING ACTIVITY** | **No of hours** |
| Lectures / Guided workshops | 12 |
| Practical classes / Support Sessions | 20 |
| **TOTAL SCHEDULED LEARNING HOURS** | 32 |
| **GUIDED INDEPENDENT STUDY** |  |
| Programming practice | 36 |
| Lecture / workshop preparation | 6 |
| Directed reading | 10 |
| Assignment Work | 16 |
| **TOTAL GUIDED INDEPENDENT STUDY HOURS** | 68 |
| ***TOTAL STUDENT LEARNING HOURS***  ***These must add up to 200 hours per 20 credits*** | 100 |

**Bibliography and Learning Support Material**

|  |
| --- |
| The most up to date bibliography for this module is available via the on-line reading list – click on the link below:  <http://readinglists.central-lancashire.ac.uk/index>  <http://cypruslists.central-lancashire.ac.uk/index.html>  Module resources on eLearn. [Online] http://bblive.uclan.ac.uk  Miles, M. (2012), *Rob Miles CSharp Yellow Book,* University of Hull (Free online book)  Sharp, J. (2010), *Microsoft Visual C# 2010 Step by Step*, Microsoft Press, ISBN-13: 978-0735626706  Visual Studio 2010 from Microsoft Academic Alliance (free to UCLan students)  or  Visual C++ 2010 Express. [Online] http://www.microsoft.com/express/Downloads/#2010-Visual-CPP |