Title:	Modular Programming APPROVED
Long Title:	Modular Programming

Module Code:	SOFT6017	**Duration:**	1 Semester

Credits:	5
NFQ Level:	Fundamental
Field of Study:	Computer Software
Valid From:	Semester 1 - 2017/18 (September 2017)
Module Delivered in	6 programme(s)
Next Review Date:	September 2021
Module Coordinator:	Sean McSweeney
Module Author:	CLIONA MC GUANE
Module Description:	This module aims to develop the student's programming knowledge. The student will learn how to separate a program into independent modules such that each module contains everything necessary to execute only one aspect of the desired functionality. Composite datatypes and classes will be employed to process data with multiple fields. The student will be introduced to the concept of version control when working on projects.

Learning Outcomes	
On successful completion of this module the learner will be able to:	
LO1	Apply modular programming to defined programming problems.
LO2	Design and develop programs that use one-dimensional arrays.
LO3	Design and develop programs that use composite data types.
LO4	Employ basic version control techniques to keep track of code revisions.

Pre-requisite learning
Incompatible Modules *These are modules which have learning outcomes that are too similar to the learning outcomes of this module. You may not earn additional credit for the same learning and therefore you may not enrol in this module if you have successfully completed any modules in the incompatible list.*
No incompatible modules listed
Co-requisite Modules
No Co-requisite modules listed
Requirements *This is prior learning (or a practical skill) that is mandatory before enrolment in this module is allowed. You may not enrol on this module if you have not acquired the learning specified in this section.*
No requirements listed
Co-requisites
No Co Requisites listed

Module Content & Assessment

Indicative Content
Modular decomposition Using modular decomposition, large problems will be broken into smaller problems. Each smaller problem may be solved using reusable pieces of code. Students will learn about arguments, parameters, return values, parameter passing mechanisms and the scope of variables in functions.
Arrays Introduction to arrays which will include, but is not limited to, creating arrays, populating arrays, processing arrays and using arrays in methods.
Composite data types and Classes Introduction to composite data types (records) for data storage, moving on to add functions to the records thus creating classes.
Version Control Students will be introduced to the benefits of version control and encouraged to use it to track changes to their code, in particular while coding their projects.

Assessment Breakdown	%
Course Work	100.00%

Course Work

Assessment Type	*Assessment Description*	*Outcome addressed*	*% of total*	*Assessment Date*
Multiple Choice Questions	This exam tests the student's knowledge of the theory of modular programming, arrays, records, classes and version control, as well as his/her ability to understand code and his/her ability to fix code.	1,2,3,4	10.0	Week 4
Practical/Skills Evaluation	An open book in-lab exam in which the student can use his/her knowledge, programs and notes to code a solution to a defined problem. A example exam would require the student to develop a programme in which the user would provide a length in metres and convert it to a length in another unit from a menu of choices e.g. feet, inches or cm.	1	15.0	Week 7
Project	The project aims to evaluate the student's programming ability by solving a defined problem. It is expected that the student be able to show the development of the code through version control. An example of a project would be a banking application. Customers could view their account details and account balances, deposit money and withdraw money (subject to specified criteria). Administrators could create accounts, delete accounts and view statistics e.g. highest deposit, smallest deposit, average deposit.	1,2,3,4	25.0	Week 10
Practical/Skills Evaluation	An open book in-lab exam in which the student can use his/her knowledge, programs and notes to code a solution to a defined problem. An example exam would ask the users to create an application to keep track of a user's contacts - the user could view his/her contacts, find a particular contact, add a contact, delete a contact and edit a contact's details e.g. first name, surname, and phone number.	1,2,3	50.0	Week 13

No End of Module Formal Examination

Reassessment Requirement
Coursework Only *This module is reassessed solely on the basis of re-submitted coursework. There is no repeat written examination.*

The institute reserves the right to alter the nature and timings of assessment

Module Workload

Workload: Full Time				
Workload Type	*Workload Description*	*Hours*	*Frequency*	*Average Weekly Learner Workload*
Lecture	Lecture delivery theory underpinning learning outcomes.	1.0	Every Week	1.00
Lab	Lab to support learning outcomes.	3.0	Every Week	3.00
Independent & Directed Learning (Non-contact)	Self study.	3.0	Every Week	3.00
			Total Hours	7.00
			Total Weekly Learner Workload	7.00
			Total Weekly Contact Hours	4.00

Workload: Part Time				
Workload Type	*Workload Description*	*Hours*	*Frequency*	*Average Weekly Learner Workload*
Lecture	Lecture delivery theory underpinning learning outcomes.	1.0	Every Week	1.00
Lab	Lab to support learning outcomes.	3.0	Every Week	3.00
Independent & Directed Learning (Non-contact)	Self study.	3.0	Every Week	3.00
			Total Hours	7.00
			Total Weekly Learner Workload	7.00
			Total Weekly Contact Hours	4.00

Module Resources

Recommended Book Resources

- **Tony Gaddis 2014, *Starting out with Python*, 3 Ed., Pearson Education Ltd. [ISBN: 9781292065502]**
- **Tony Gaddis 2015, *Start out with Java: From Control Structures to Objects*, 6 Ed., Pearson Education Ltd. [ISBN: 9781292110653]**

This module does not have any article/paper resources

Other Resources

- **Website: *YouTube video tutorials for a variety of programming languages***
  http://www.thenewboston.com/
- **Website: *Python Style Guide***
  https://www.python.org/dev/peps/pep-0008 /
- **Website: *Python IDE***
  https://www.jetbrains.com/pycharm/
- **Website: *Java IDE***
  http://www.bluej.org/
- **Website: *IDE***
  https://eclipse.org/
- **Website (Version Control): *GitHub (Git Solution)***
  https://github.com/
- **Website (Version Control): *Git***
  https://git-scm.com/
- **Website (Version Control): *BitBucket (Git Solution)***
  https://bitbucket.org/
- **Website: *Python Language Reference***
  https://docs.python.org/3/
- **Website: *Java JDK***
  http://www.oracle.com/technetwork/java/j avase/downloads/index.html

Module Delivered in

Programme Code	Programme	Semester	Delivery
CR_KSDEV_8	Bachelor of Science (Honours) in Software Development	2	Mandatory
CR_KDNET_8	Bachelor of Science (Honours) in Computer Systems	2	Mandatory
CR_KITMN_8	Bachelor of Science (Honours) in IT Management	2	Mandatory
CR_KITSP_7	Bachelor of Science in Information Technology	2	Mandatory
CR_KCOMP_7	Bachelor of Science in Software Development	2	Mandatory
CR_KCOME_6	Higher Certificate in Science in Software Development	2	Mandatory