MODULE DESCRIPTOR

Module Title

Concurrent Programming

Reference	CM3113	Version	1
Created	March 2017	SCQF Level	SCQF 9
Approved	April 2005	SCQF Points	15
Amended	April 2017	ECTS Points	7.5

Aims of Module

To provide the student with knowledge of concurrent programming techniques and to develop the student's ability to design, implement and verify effective and secure software solutions within a concurrent programming development environment.

Learning Outcomes for Module

On completion of this module, students are expected to be able to:

1	Analyse user requirements and develop a concurrent solution as a collection of interacting threads of execution.
2	Identify concurrent interactions within the overall design and select an appropriate combination of synchronisation mechanisms to handle these interactions.
3	Apply analytic rigour to verify correctness of an overall design approach.
4	Implement the design in a concurrent programming environment, making a critical selection of the facilities that provide support for multi-threading and synchronisation.

Indicative Module Content

Key concepts of multi-threaded programming including: thread attributes, thread life history, scheduling. Indivisible operations, race conditions, safety and liveness, formal approaches to verifying correctness of a concurrent design. Synchronisation primitives based on use of: shared variables, test and set primitives, semaphores, monitors. Generic concurrent programming problems and their solution: mutual exclusion, resource allocation, event ordering, interthread communication. Client Server systems and secure connections using Java Sockets. Security issues in concurrent systems.

Module Delivery

Key concepts are introduced and illustrated through the medium of lectures. Self-paced tutorial questions and solutions are made available via the Virtual Learning Environment. In the accompanying laboratory sessions the student will progress through a series of design and implementation exercises intended to test the student's understanding of the lecture content and to develop proficiency in the practical application of concurrent programming skills.

Module Ref:	CM3113 v1

Indicative Student Workload	Full Time	Part Time
Contact Hours	44	N/A
Non-Contact Hours	106	N/A
Placement/Work-Based Learning Experience [Notional] Hours	N/A	N/A
TOTAL	150	N/A
Actual Placement hours for professional, statutory or regulatory body		

ASSESSMENT PLAN

If a major/minor model is used and box is ticked, % weightings below are indicative only.

Component 1

Type:	Examination	Weighting:	50%	Outcomes Assessed:	2, 3
Description:	This is a closed book examination worth 50% of the total module assessment.				

Component 2

Type:	Coursework	Weighting:	50%	Outcomes Assessed:	1, 4
Description:	This is a coursework assignment worth 50% of the total module assessment.				

MODULE PERFORMANCE DESCRIPTOR

Explanatory Text

The calculation of the overall grade for this module is based on equal weighting of C1 and C2. The minimum grade required to obtain a pass is D.

		Examination:						
		A	**B**	**C**	**D**	**E**	**F**	**NS**
Coursework:	**A**	A	A	B	B	C	E	
	B	A	B	B	C	C	E	
	C	B	B	C	C	D	E	
	D	B	C	C	D	D	E	
	E	C	C	D	D	E	E	
	F	E	E	E	E	E	F	
	NS	Non-submission of work by published deadline or non-attendance for examination						

Module Requirements

Prerequisites for Module	CM2100 Advanced Software Design and Development or equivalent.
Corequisites for module	None.
Precluded Modules	None.

Module Ref:	CM3113 v1

INDICATIVE BIBLIOGRAPHY

1	GOETZ Brian., 2006. Java Concurrency in Practice. Addison-Wesley.
2	GONZALEZ Javier F., 2016. Mastering Concurrency Programming with Java 8. Packt Publishing.
3	FREISEN Jeff, 2015. Java Threads and the Concurrency Utilities. Apress.
4	The Java Tutorials: Oracle Java Documentation available via https://docs.oracle.com/javase/tutorial/essential/concurrency/