


## MODULE SYLLABUS

### PROGRAMMING METHODOLOGY AND DESIGN

<b>MODULE CODE</b>	CCS1115
<b>MODULE TITLE</b>	Programming Methodology and Design
<b>PROGRAMME</b>	BSc (Hons) in Computer Science
<b>DEPARTMENT</b>	Computer Science
<b>CREDITS</b>	10
<b>STAGE OF STUDY</b>	1
<b>SEMESTER/SESSION</b>	Fall 2021-2022
<b>RE-ASSESSABLE</b>	YES
<b>COMPENSATABLE</b>	YES
<b>LOCATION</b>	Thessaloniki
<b>STAFF</b>	Dr Ioanna Stamatopoulou
<b>E-MAIL</b>	<a href="mailto:istamatopoulou@york.citycollege.eu">istamatopoulou@york.citycollege.eu</a>
<b>STAFF OFFICE</b>	L. Sofou bldg., 6 <sup>th</sup> floor
<b>ACCREDITATION</b>	The programme is accredited by: the British Computer Society (BCS) 

<b>DESCRIPTION</b>
This module is a continuation of the module Programming Principles and Algorithms, introducing students to further fundamental constructs and practices required for the procedural development of software in the Java programming language. Emphasis is placed on problem analysis, developing algorithmic thinking skills, and testing one's code.

<b>AIMS</b>
This module aims to:
A1 fundamental programming concepts and techniques; and
A2 testing Java functions.

LEARNING OUTCOMES		
By the end of the module, a student will be able to:		Link to aims
LO1	define user-defined functions and call them;	A1
LO2	make use of arrays when necessary;	A1
LO3	break a problem into smaller parts; and	A1
LO4	test their Java functions using JUnit.	A2

<b>HOW DOES THIS MODULE FIT INTO THE CURRICULUM?</b>
This module builds on the foundation for algorithmic thinking, problem-solving, and procedural programming, which is necessary for any computer scientist, to introduce students to functions. The principles learned are applicable to any other high-level programming language students may learn in the future and will also be used when students will be introduced in Object-Oriented Programming in the next semester. Finally, Java, as the actual language learnt, will be useful for a number of future modules, such as Data Structures and Algorithms in Stage 2.

<b>TEACHING &amp; LEARNING METHODS</b>	Total Contact Hours: 22
The following teaching & learning methods will be employed: The module is delivered through two 2-hour sessions every week.	
<ul style="list-style-type: none"> <li>During the first 2 hours the theoretical concepts are taught and students have the opportunity to immediately work on small example/abstract exercises so that they practice the Java syntax.</li> <li>During the second 2 hours students work entirely on lab exercises that build up in complexity, so that they develop their algorithmic thinking and problem solving skills.</li> </ul>	

ASSESSMENT METHODS				
#	Students will be assessed by:	Submission Week	% contribution	LOs assessed
1	Project: group, one-day challenge	RA13	35%	LO1 – LO3
2	Assessed Lab	RA15	65%	LO1, LO2, LO4

FEEDBACK PROVISION				
<p>The following methods will be used to provide feedback to students:</p> <ul style="list-style-type: none"> <li>Formative feedback: Students have the opportunity to receive formative (group as well as individual) feedback every week during the labs as they get to work alone or in pairs on given exercises, before we solve them together.</li> <li>Summative feedback will be provided to the entire class and per group after the submission of the project.</li> </ul> <p>The feedback handbook found at <a href="https://goo.gl/Zy2roA">https://goo.gl/Zy2roA</a> aims to give you a better understanding of feedback; what it is for and how to use it.</p>				

ACCESS TO MODULE MATERIAL (Notes, handouts, announcements etc.)				
All material used in this module's classes are available in electronic form through Google Classroom.				

RECOMMENDED TEXTBOOK(S)				
Paul Deitel, Harvey Deitel, <i>Java How To Program (late objects)</i> , 10 <sup>th</sup> edition, Prentice Hall, 2015.				

LIST OF REFERENCES / ADDITIONAL RECOMMENDED READING				
Brian P. Hogan, <i>Exercises for programmers: 57 challenges to develop your coding skills</i> , The Pragmatic Bookshelf, Dallas, Texas, 2015.				

OUTLINE	
WEEK	
#7	<b>Introduction to Functions</b> Function Signature Definition
#8 - #9	<b>Functions (cont'd)</b> Implementation and Calling <b>Testing with JUnit</b>
#10 - #11	<b>Arrays</b> Declaration and use, 2-dimensional arrays Arrays as parameters to functions, Pass-by-Value vs. Pass-by-Reference <b>Java Documentation</b>
#12	<b>Revision week</b>

EMPLOYABILITY PROFILE	
<p>This module contributes to your employability profile by enhancing the following Graduate Attributes:</p> 