Module Specification

1. Factual information			
Module title	**M269: Algorithms, Data structures and Computability.**	**Level**	**2**
Module tutor	TBA	**Credit value**	**30**
Module type	Taught	**Notional learning hours**	**8**

2. Rationale for the module and its links with other modules
One of the basic pillars of advanced computing projects consists of the set of proper algorithms used to solve not only traditional but also unconventional IT problems. With the huge amount of data embedding the new data science, being skilled in setting proper data structure, managing and understanding computability techniques become a must nowadays. M269 is one of the most important modules for information technologies and computing related majors and tracks. The underlying concepts of this module are implemented using the python programming language.

3. Aims of the module
This module aims to • Provide the students with the required skills to possess the computational thinking. These skills start by proper understanding and analyzing the problems to be solved and end by providing computer programs that solve these problems. • One of the important aspects of this module is to provide the students with the awareness of the limits of computation and the ability to decide which problems can and which cannot be solved efficiently with computers.

4. Pre-requisite modules or specified entry requirements
This module is offered in 4 tracks: WD, CS, ITC and CwB. Studying this module requires a certain basic knowledge in prgramming and maths. Pre-requisites are TM105 & MT131

5. Intended learning outcomes	
A. Knowledge and understanding	**Learning and teaching strategy**
Upon completing this module, students will be able to: **A1.** Identify and define the sets, functions and logic, and their application in the design, implementation and analysis of computer-based systems. **A2.** Define and recognize Data structure and computational problematic.	• 25% face-to-face tutorial sessions • TMA work • Module learning booklets and support material • Support material on LMS

B. Cognitive skills	**Learning and teaching strategy**
Upon completing this module, students will be able to: **B1.** Explain, construct and use algorithms and data structures to solve computational problems. **B2.** Describe and assess the difficulty of computational problems. **B3.** Analyse algorithms and computational problems making use of several informal proof techniques	• 25% face-to-face tutorial sessions • TMA work • Module learning booklets and support material • Support material on LMS

C. Practical and professional skills	**Learning and teaching strategy**
Upon completing this module, students will be able to: **C1.** Use the Python programming language to implement algorithms.	• 25% face-to-face tutorial sessions • TMA work

C. Practical and professional skills	Learning and teaching strategy
C2. Write a short report which is based on one or more sources and which has a well-argued conclusion.	• Module learning booklets and support material • Support material on LMS

D Key transferable skills	Learning and teaching strategy
Upon completing this module, students will be able to: **D1.** Apply appropriate computational problem-solving techniques to a range of problems; **D2.** Apply computational thinking skills to solve problems across a range of application areas. **D3.** Discuss the questions 'What is computation?' and 'What are its limits?', and explain how the answers to these questions have important implications for the practical use of computer-based systems.	• 25% face-to-face tutorial sessions • TMA work • Module learning booklets and support material • Support material on LMS

6. Indicative content.		
Unit 1	**Introduction** Introduction What is computation? Introducing Python Introduction Come fly with Python Basic Python Why Python? Computational thinking	

6. Indicative content.

Unit 2	**From problems to programs** From problem to program Getting the inputs and outputs Getting the algorithm Getting the ADT A taste of formal logic Iteration and logic Pre- and post-conditions Correctness and clarity Getting data structures right How do we know it is right? Dividing and conquering
Unit 3	**Sorting** What is sorting? Naive sorting Bubble sort Selection sort Insertion sort Complexity of straight sorting algorithms Inducing, reducing and recusing Induction Reduction and recursion Recursive sorting Sorting smart Dividing and conquering Trees and heaps Sorting – two final thoughts
Unit 4	**Searching** Searching lists Searching for patterns

6. Indicative content.	
	Basic string search The quick search algorithm The Knuth–Morris–Pratt algorithm Other algorithms Maps Hashing and hash tables Search trees Binary search trees AVL trees
Unit 5	**Optimisation** Optimisation and optimisation problems Graphs and greed Dynamic programming
Unit 6	**Sets, logic and databases** Sets and propositional logic Predicate logic, or first order logic Database retrieval using simple queries The cardinality of infinite sets
Unit 7	**The limits of computation** Computability Logic revisited Computational complexity Physics and computing

7. Assessment strategy, assessment methods and their relative weightings
TMA Work: 20% MTA: 30% Exam: 50%

8. Mapping of assessment tasks to learning outcomes										
Assessments tasks	**Learning Outcomes**									
	A1	**A2**	**B1**	**B2**	**B3**	**C1**	**C2**	**D1**	**D2**	**D3**
TMA	✓	✓	✓		✓	✓		✓	✓	
MTA	✓	✓		✓	✓	✓			✓	
Final	✓		✓	✓		✓	✓	✓	✓	✓

9. Teaching staff associated with the module	
Tutor's name and contact details	**Contact hours**
Dr. Nouhad Amaneddine **Dr. Radwan Abu Jassar** **Mr. Ahmad Mikati**	

10. Key reading list				
Author	**Year**	**Title**	**Publisher**	**Location**
Module adopted from OU, UK.				
The Open University	**2015**	**Logic and the limit of computing**	**The open university**	
Magnus Lie Hetland	**2010**	**Python Algorithms: Mastering Basic Algorithms in the Python Language**	**Apress**	
Allen Downey	**2012**	**Think Python**	**Green Tea Press**	

11. Other indicative text (e.g. websites)
https://lms.arabou.edu.kw