1. **Title of the module**

COMP5180 (CO518) - Algorithms, Correctness & Efficiency

1. **Division or partner institution which will be responsible for management of the module**

Division of Computing, Engineering, Mathematical Sciences (CEMS)

1. **The level of the module (Level 4, Level 5, Level 6 or Level 7)**

Level 5

1. **The number of credits and the ECTS value which the module represents**

15 credits (7.5 ECTS)

1. **Which term(s) the module is to be taught in (or other teaching pattern)**

Autumn or Spring

1. **Prerequisite and co-requisite modules**

Pre-requisite: COMP3830: Problem Solving with Algorithms

COMP5200: Further Object-Oriented Programming

COMP3250: Foundations of Computing II

1. **The courses of study to which the module contributes**

BSc Computer Science and all variants, BSc Computing, BSc Software Engineering with a year in industry variants of these courses

1. **The intended subject specific learning outcomes.  
   On successfully completing the module students will be able to:**

8.1 Specify, test, and verify implementations of algorithms;

8.2 Analyse the time and space behaviour of algorithms;

8.3 Analyse and compare general algorithmic paradigms;

8.4 Make informed decisions while choosing data structures and algorithms for practical use;

8.5 Demonstrate an understanding of algorithmic reduction, complexity classes and hardness.

1. **The intended generic learning outcomes.  
   On successfully completing the module students will be able to:**

9.1 Demonstrate an understanding of trade-offs when making design decisions;

9.2 Make effective use of existing techniques to solve problems;

9.3 Demonstrate an understanding of the mathematical underpinnings of algorithmic correctness and efficiency analysis;

9.4 Analyse and compare solutions to algorithmic problems.

1. **A synopsis of the curriculum**

The curriculum covers topics in algorithms and data structures. Among data structures, it covers advanced topics on trees, heaps, graphs, et cetera. It provides details of computational complexity notations like O(). It covers the correctness and runtime analysis of recursive algorithms using recurrences. These algorithms range from mathematical computations to sorting algorithms. These algorithms are put in the context of appropriate algorithmic paradigms like divide-and-conquer and dynamic programming. Finally, computational complexity classes and problem reductions are introduced along with the proof techniques for NP-hardness and NP-completeness.

1. **Reading list (Indicative list, current at time of publication. Reading lists will be published annually)**

## The University is committed to ensuring that core reading materials are in accessible electronic format in line with the Kent Inclusive Practices.

## The most up to date reading list for each module can be found on the university's [reading list pages](https://kent.rl.talis.com/index.html).

Algorithms. Sedgewick and Wayne

Algorithms. Jeff Erickson

Introduction to Algorithms. Cormen, Leiserson, Rivest, and Stein

The Art of Computer Programming. Donald E. Knuth

The Algorithm Design Manual. Steven S. Skiena

Data Structures and Algorithms in Java 2nd Edition. M.T. Goodrich and R. Tamassia

Algorithms and Data Structures 2nd Edition. Jeffrey H. Kingston x

Cracking the Coding Interview. Gayle Laakmann McDowell

1. **Learning and teaching methods**

Total contact hours: 32

Private study hours: 118

Total study hours: 150

1. **Assessment methods**
   1. Main assessment methods

2 programming assessments (15 hours each) (25% each)

2 hour unseen written examination (50%)

13.2 Reassessment methods

Like for like.

1. **Map of module learning outcomes (sections 8 & 9) to learning and teaching methods (section12) and methods of assessment (section 13)**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Module learning outcome** | *8.1* | *8.2* | *8.3* | *8.4* | 8.5 | *9.1* | *9.2* | *9.3* | *9.4* |
| **Lectures** | x | x | x | x | x | x | x | x | x |
| **Private Study** | x | x | x | x | x | x | x | x | x |
| **Assessment method** |  |  |  |  |  |  |  |  |  |
| *Exam* |  | x | x | x | x | x | x | x | x |
| *Coursework* | x |  |  | x |  | x | x |  |  |

1. **Inclusive module design**

The Division recognises and has embedded the expectations of current equality legislation, by ensuring that the module is as accessible as possible by design. Additional alternative arrangements for students with Inclusive Learning Plans (ILPs)/declared disabilities will be made on an individual basis, in consultation with the relevant policies and support services.

The inclusive practices in the guidance (see Annex B Appendix A) have been considered in order to support all students in the following areas:

a) Accessible resources and curriculum

b) Learning, teaching and assessment methods

1. **Campus(es) or centre(s) where module will be delivered**

Canterbury

1. **Internationalisation**

The topics addressed by this module relate to a field which is of international importance, given the global role of computers in today's technological innovation. The topics covered by this module are international in nature, being identical worldwide and independent of traditional spoken language.

**DIVISIONAL USE ONLY**

**Revision record – all revisions must be recorded in the grid and full details of the change retained in the appropriate committee records.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Date approved | Major/minor revision | Start date of delivery of revised version | Section revised | Impacts PLOs (Q6&7 cover sheet) |
| 23/11/2020 | Minor | September 2021 | 7 | No |
| 01/12/2021 | Major | Sept 2022 | 6, 7, 8, 9, 10, 11 & 12 | No |