<table>
<thead>
<tr>
<th>Module Code</th>
<th>CSU22010</th>
</tr>
</thead>
<tbody>
<tr>
<td>Module Name</td>
<td>Algorithms and Data Structures</td>
</tr>
<tr>
<td>ECTS Weighting</td>
<td>10 ECTS</td>
</tr>
<tr>
<td>Semester taught</td>
<td>Semester 1 &amp; 2</td>
</tr>
<tr>
<td>Module Coordinator/s</td>
<td>Dr. Ivana Dusparic, Dr. Vasileios Koutavas</td>
</tr>
</tbody>
</table>

**Module Learning Outcomes**

On successful completion of this module, students will:

- **LO1.** Have gained significant knowledge on algorithms and data structures, and the mathematical theory and techniques to evaluate their efficiency and effectiveness.
- **LO2.** Have the ability to evaluate algorithms in terms of their running time and memory space requirements and classify those algorithms in the major complexity classes using appropriate performance models.
- **LO3.** Be able to efficiently implement the operations of the main data structures used in most programming.
- **LO4.** Have gained experience through experiments in implementing effective new and existing algorithms.
- **LO5.** Be able to identify the most suitable data structures and algorithms for each programming problem based on the parameters of the problem, the advantages and limitations of each data structure and algorithm, the resources available, the desired performance criteria etc.
- **LO6.** Be able to design and implement robust, effective and well-structured Java programs using industry standards such as Abstract Data Types and the approaches of unit testing, test coverage, Design by Contract, and pre-/post-conditions. Students will be able to use the last two approaches to avoid defensive programming.

**Module Content**

**Theory:**

- Asymptotic growth functions and analysis of source code to derive running time and space requirements
- Amortised running time analysis of algorithms
- Permutations, Combinations and Sets

**Data structures:**

- Array and linked list implementations of stacks and queues.
- Doubly linked lists
- Union-find
- Binary trees, binary search trees, balanced search trees, B-trees
- Hash tables
- Undirected, directed and weighted graph implementations using adjacency lists

1. [TEP Glossary](#)
- Special topics

**Algorithms:**
- Recursion vs iteration; tree traversals
- Greedy algorithms
- Divide and conquer
- Graph algorithms
- Searching and Sorting algorithms
- Dynamic Programming algorithms
- Special topics

**Programming:**
- Java generics
- Iterators
- JUnit testing

---

**Teaching and Learning Methods**

3 hours of lectures, 1 hour of laboratories per week. Individual coursework assignments. In-class quizzes and tests.

---

### Assessment Details

<table>
<thead>
<tr>
<th>Assessment Component</th>
<th>Brief Description</th>
<th>Learning Outcomes Addressed</th>
<th>% of total</th>
<th>Week set</th>
<th>Week due</th>
</tr>
</thead>
<tbody>
<tr>
<td>Examination</td>
<td>2 hour written examination</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td>50%</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>Assignment S1-1</td>
<td>Introductory in-lab Assignment 1</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td>2.625%</td>
<td>Week 1</td>
<td>Week 1</td>
</tr>
<tr>
<td>Assignment S1-2</td>
<td>Assignment 2</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td>7.4375%</td>
<td>Week 3</td>
<td>Week 6</td>
</tr>
<tr>
<td>Assignment S1-3</td>
<td>Assignment 3</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td>7.4375%</td>
<td>Week 9</td>
<td>Week 12</td>
</tr>
<tr>
<td>Assignment S2-1</td>
<td>Assignment 4</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td>8.75%</td>
<td>Week 14</td>
<td>Week 18</td>
</tr>
<tr>
<td>Assignment S2-2</td>
<td>Assignment 5</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td>8.75%</td>
<td>Week 21</td>
<td>Week 24</td>
</tr>
<tr>
<td>S1 E-Test</td>
<td>Semester 1 in-class e-test</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td>7.5%</td>
<td>Week 8</td>
<td>Week 8</td>
</tr>
<tr>
<td>S2 E-Test</td>
<td>Semester 2 in-class e-test</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td>7.5%</td>
<td>Week 20</td>
<td>Week 20</td>
</tr>
<tr>
<td>Bonus Assignment</td>
<td>Bonus Assignment</td>
<td>LO1, LO2, LO3, LO4, LO5, LO6</td>
<td></td>
<td>Week 1</td>
<td>Week 13</td>
</tr>
</tbody>
</table>

---

**Reassessment Details**

Examination (2 hours, 100%)

---

2 TEP Guidelines on Workload and Assessment
### Contact Hours and Indicative Student Workload

<table>
<thead>
<tr>
<th>Contact Hours (scheduled hours per student over full module), broken down by:</th>
<th>88 hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>lecture</td>
<td>66 hours</td>
</tr>
<tr>
<td>laboratory</td>
<td>22 hours</td>
</tr>
<tr>
<td>tutorial or seminar</td>
<td>0 hours</td>
</tr>
<tr>
<td>other</td>
<td>0 hours</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Independent study (outside scheduled contact hours), broken down by:</th>
<th>134 hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>preparation for classes and review of material (including preparation for examination, if applicable)</td>
<td>72 hours</td>
</tr>
<tr>
<td>completion of assessments (including examination, if applicable)</td>
<td>62 hours</td>
</tr>
</tbody>
</table>

| Total Hours | 222 hours |

### Recommended Reading List

**Main textbook:**

*Algorithms (4th Edition)*
Robert Sedgewick and Kevin Wayne
Pearson Education 2011

**Other recommended textbooks:**

*Introduction to Programming Using Java (6th Edition)*
David J. Eck
[http://math.hws.edu/javanotes/](http://math.hws.edu/javanotes/)

*Introduction to Algorithms (3rd Edition)*
Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein
[http://mitpress.mit.edu/books/introduction-algorithms](http://mitpress.mit.edu/books/introduction-algorithms)
MIT Press 2009

### Module Pre-requisites

**Prerequisite modules:** list module codes

**Other/alternative non-module prerequisites:** Programming in Java

### Module Website

[Module Website](#)

### Last Update

17/6/2019 by Vasileios Koutavas & Ivana Dusparic