Data structures and Algorithms

1. **Course number and name:** 020SDAES3/020DSAES3 Data structures and Algorithms

2. **Credits and contact hours:** 4 ECTS credits, 2x1:15 contact hours

3. **Name of course coordinator:** Dany Mezher

4. **Instructional materials:** Course handouts; slides; in-class problems, E-learning support using Moodle

   References:
   - Robert Sedgewick, Algorithms in C++, 3rd Edition.

5. **Specific course information**
   a. **Catalog description:**
   Complexity analysis, Elementary data structures (Arrays, Linked lists, stacks, queues), Search problems (sequential search, bisection), Sorting (elementary sorts, quicksort, merge sort), trees (characteristics, structure, traversal), string search algorithms, priority queues, heap, graphs (characteristics and structure), graph algorithms (shortest path, spanning tree, connectivity…), , scheduling problems, flow problems (maximum flow, minimum cost flow problem,…), coupling, dynamic programming.

   b. **Prerequisites:** None

   c. **Required** for CCE students

6. **Educational objectives for the course**
   a. **Specific outcomes of instruction:**
   - Choose the appropriate algorithms and data structures to use when implementing a solution.
   - Analyze the complexity of a solution.
   - Use professional tools to debug, profile and test the solution

   b. **PI addressed by the course:**

PI	1.1	1.2	1.3	2.4	2.5	6.3	6.4	7.1
Covered	x	x	x	x	x	x	x	x
Assessed		x	x	x		x	x	

7. **Brief list of topics to be covered**

   - Professional development tools: Source code management, build chain, build options, profiling, documenting (1 lecture)

- Complexity analysis, elementary data structures (Linked lists, Arrays, Stacks and Queues) (1 lecture)
- Implement basic data structures, debugging, Unit Testing
- Deliverables git repository: source code + documentation (1 lab session)
- Search problems: sequential search, bisection (1 lecutre)
- Sorting: Elementary sorting algorithms, Quicksort, Merge Sort (1 lecture)
- Algorithm implementation, performance analysis (unit testing, debug vs release, profiling) Deliverables git repository: source code + documentation (1 lab session)
- Recursive vs Iterative algorithms (1 lecture)
- General purpose algorithms: strings algorithms, graphic algorithms (1 lecture)
- Recursive algorithms, performace analysis, unit testing, profiling
- Deliverables git repository: source code + documentation (1 lab session)
- Trees: Characteristics, structure and traversal (1 lecture)
- Priority queues, heap (1 lecture)
- Algorithm implementation, performance analysis (unit testing, debug vs release, profiling) Deliverables git repository: source code + documentation (1 lab session)
- Graphs: characteristics, structure (1 lecture)
- Graphs traversal: Depth First search, Breadth first search (1 lecture)
- Algorithm implementation, performance analysis (unit testing, debug vs release, profiling) Deliverables git repository: source code + documentation (1 lab session)
- Graph algorithms: union-find, connectivity, spanning tree, shortest path (2 lectures)
- Algorithm implementation, performance analysis (unit testing, debug vs release, profiling) Deliverables git repository: source code + documentation (1 lab session)
- Dynamic programming (2 lectures)
- Algorithm implementation, performance analysis (unit testing, debug vs release, profiling) Deliverables git repository: source code + documentation (1 lab session)