COURSE OUTLINE

(1) GENERAL

SCHOOL	ENGINEERING		
ACADEMIC UNIT	Department of Computer Engineering and Informatics		
LEVEL OF STUDIES	Undergraduate		
COURSE CODE	**CEID_NY205**	**SEMESTER**	**Winter**
COURSE TITLE	Introduction to Algorithms		

INDEPENDENT TEACHING ACTIVITIES *if credits are awarded for separate components of the course, e.g. lectures, laboratory exercises, etc. If the credits are awarded for the whole of the course, give the weekly teaching hours and the total credits*	**WEEKLY TEACHING HOURS**	**CREDITS**
Lectures, Tutorials, Laboratory	2(L), 2(T), 2(Lab)	6
Add rows if necessary. The organisation of teaching and the teaching methods used are described in detail at (d).	TOTAL	6

COURSE TYPE *general background, special background, specialised general knowledge, skills development*	Special background
PREREQUISITE COURSES:	Recommended prerequisite knowledge: "Discrete Mathematics" (NY109), or equivalent.
LANGUAGE OF INSTRUCTION and EXAMINATIONS:	Greek (English if there are Erasmus students)
IS THE COURSE OFFERED TO ERASMUS STUDENTS	Yes
COURSE WEBSITE (URL)	https://www.ceid.upatras.gr/webpages/faculty/zaro/teaching/intro-alg/index.html

(2) LEARNING OUTCOMES

Learning outcomes

The course learning outcomes, specific knowledge, skills and competences of an appropriate level, which the students will acquire with the successful completion of the course are described.

Consult Appendix A

- *Description of the level of learning outcomes for each qualifications cycle, according to the Qualifications Framework of the European Higher Education Area*
- *Descriptors for Levels 6, 7 & 8 of the European Qualifications Framework for Lifelong Learning and Appendix B*
- *Guidelines for writing Learning Outcomes*

Upon conclusion of the course the students ought to be able to:

- Understand fundamental algorithmic concepts and techniques.
- Apply basic techniques for the solution of fundamental algorithmic problems.
- Apply basic analysis methods for determining the complexity of algorithms.
- Apply basic mathematical methods for determining the correctness of algorithms.
- Understand how to implement efficiently the algorithms taught, and to how to tackle practical issues encountered during implementation.

Upon conclusion of the course the students are expected to have the following skills/competences:

- Abstracting the core algorithmic sub-problems from given complex problems.
- Use basic techniques for designing algorithms for fundamental and more complex problems.
- Use basic methods for analyzing the complexity and correctness of algorithms.
- Implement efficiently fundamental algorithms using basic techniques and data structures.

General Competences

Taking into consideration the general competences that the degree-holder must acquire (as these appear in the Diploma Supplement and appear below), at which of the following does the course aim?

Search for, analysis and synthesis of data and information, Project planning and management

with the use of the necessary technology *Adapting to new situations* *Decision-making* *Working independently* *Team work* *Working in an international environment* *Working in an interdisciplinary environment* *Production of new research ideas*	*Respect for difference and multiculturalism* *Respect for the natural environment* *Showing social, professional and ethical responsibility and sensitivity to gender issues* *Criticism and self-criticism* *Production of free, creative and inductive thinking* *……* *Others…* *……*

Search for, analysis and synthesis of data and information, with the use of the necessary technology
Adapting to new situations
Decision-making
Working independently
Criticism and self-criticism
Production of free, creative and inductive thinking

(3) SYLLABUS

1. ***Elementary Concepts in the Design and Analysis of Algorithms***
   The concept of algorithm, applications and importance of algorithms. The concept of efficiency, a model for measuring efficiency, methods for analyzing the complexity of algorithms, technological importance of efficient algorithms.
2. ***Basic Concepts in the Analysis and Complexity of Algorithms***
   Efficiency and time complexity, optimal algorithms, methods in analyzing the complexity of algorithms, asymptotic complexity, correctness of algorithms.
3. ***Elementary Algorithms and Data Structures***
   Arrays, lists, stacks, queues, trees. Algorithms for finding the minimum or maximum in a set of elements. Algorithms for merging sorted lists, insertion sort, binary search, counting element tuples. Heap, priority queues, and their application in sorting (heapsort).
4. ***Stable Matching***
   Problem formulation and applications. The propose-and-reject algorithm. Correctness and complexity analysis of the algorithm. Efficient implementation of the algorithm.
5. ***The Divide-and-Conquer Technique***
   Generic description of the divide-and-conquer technique. The merge-sort algorithm. The algorithm for counting inversions. Recurrence relations and methods for their solution.
6. ***Graphs and Graph Algorithms***
   Graphs as fundamental model of networks and systems. Basic properties and features of graphs. Graph connectivity. Graph traversal and searching algorithms: breadth-first-search (BFS), depth-first-search (DFS). Extensions/applications of BFS and DFS for computing connected components, topological sorting, strongly connected components, and for checking graph bipartiteness.
7. ***The Greed Technique***
   Generic description of the greed technique. Scheduling algorithms: interval scheduling, scheduling all intervals, scheduling to minimize lateness. Network optimization algorithms: minimum spanning tree (Kruskal's and Prim's algorithms), shortest paths (Dijkstra's algorithms). Efficient implementation of network optimization algorithms.
8. ***The Dynamic Programming Technique***
   Generic description of the dynamic programming technique. Efficient application and implementation of dynamic programming. Algorithms for weighted interval scheduling and knapsack problems.

(4) TEACHING and LEARNING METHODS - EVALUATION

DELIVERY *Face-to-face, Distance learning, etc.*	Face-to-face. Tutorials and laboratory sessions with exemplary solutions of exercises.
USE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY *Use of ICT in teaching, laboratory education, communication with students*	ICT methods are used in both teaching and communication with the students. Lecture slides and supplementary material are uploaded in the course's web site.
TEACHING METHODS *The manner and methods of teaching are described in detail.* *Lectures, seminars, laboratory practice, fieldwork, study and analysis of bibliography, tutorials, placements, clinical practice, art workshop, interactive teaching, educational visits, project, essay writing, artistic creativity, etc.*	

Activity	***Semester workload***
Lectures	2*13=26
Tutorials (exercises)	2*13=26
Laboratory practice	2*13=26
Individual study, preparation and problem solving	3*13=39

The student's study hours for each learning activity are given as well as the hours of non-directed study according to the principles of the ECTS	Weekend study	2*13=26
	Mid-term exam preparation (1 week)	5*1=5
	Study during the 3 "empty weeks" (2 weeks of vacation and 1 week of exam preparation)	5*3=15
	Course total (25-30 hours per ECTS unit)	**163**

STUDENT PERFORMANCE EVALUATION *Description of the evaluation procedure* *Language of evaluation, methods of evaluation, summative or conclusive, multiple choice questionnaires, short-answer questions, open-ended questions, problem solving, written work, essay/report, oral examination, public presentation, laboratory work, clinical examination of patient, art interpretation, other* *Specifically-defined evaluation criteria are given, and if and where they are accessible to students.*	The language of instruction and examination is Greek. Special provisions (lecture notes and examinations in English) can be made for foreign students. Evaluation (criteria can be found in the web site of the course): • Mid-term written examination (40% of final mark). • Final written examination (60% of final mark). Written examination (mid-term and final): graded difficulty, including short-answer questions, algorithm design for problem solving, proofs of algorithm correctness and complexity, exercises. Series of laboratory (practical and theoretical) exercises aiming at familiarizing students with: • Efficient implementation of algorithms in C++ using software algorithmic platforms and libraries (e.g, LEDA, Boost). • The use of the algorithmic techniques taught in the course. • Solving algorithmic problems in practice as well as in interpreting and evaluating the results obtained.

(4) ATTACHED BIBLIOGRAPHY

- *Suggested bibliography:*
  - J. Kleinberg and E. Tardos, *Algorithm Design*, Pearson Addison-Wesley, 2006.
  - T. Cormen, C. Leiserson, R. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd Edition, MIT Press, 2009.
  - K. Mehlhorn and P. Sanders, *Algorithms and Data Structures – The Basic Toolbox*, Springer, 2008.
  - Lecture notes and slides uploaded in the web site of the course.
- *Related academic journals:*
  - This is an introductory course. Hence, there is no systematic use of articles from the scientific literature, even though presentations make reference to the recent literature mostly to demonstrate to students the relevance of the course for the state of the art in Computer Science and Engineering and their applications.