COURSE OUTLINE

(1) GENERAL

<table>
<tr><td>SCHOOL</td><td colspan="3">SCHOOL OF ENGINEERING</td></tr>
<tr><td>ACADEMIC UNIT</td><td colspan="3">COMPUTER ENGINEERING AND INFORMATICS</td></tr>
<tr><td>LEVEL OF STUDIES</td><td colspan="3">UNDERGRADUATE</td></tr>
<tr><td>COURSE CODE</td><td>NY302</td><td>SEMESTER</td><td>6th</td></tr>
<tr><td>COURSE TITLE</td><td colspan="3">COMPUTATIONAL COMPLEXITY</td></tr>
</table>

<table>
<tr><td>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</td><td>WEEKLY TEACHING HOURS</td><td>CREDITS</td></tr>
<tr><td>Lectures, Recitation sections</td><td>(2)X(2)</td><td>4</td></tr>
<tr><td>Add rows if necessary. The organisation of teaching and the teaching methods used are described in detail at (d).</td><td>TOTAL</td><td>4</td></tr>
</table>

<table>
<tr><td>COURSE TYPE
general background, special background, specialised general knowledge, skills development</td><td>General background, specialised general knowledge, skills development</td></tr>
<tr><td>PREREQUISITE COURSES:</td><td>There are no prerequisite courses. It is however recommended that students have at least a basic mathematical background, and prior involvement with the courses “Discrete Mathematics” (NY109), “Graph Theory and Applications” (NY202), “Introduction to Algorithms” (NY205), and “Theory of Computation” (NY205)</td></tr>
<tr><td>LANGUAGE OF INSTRUCTION and EXAMINATIONS:</td><td>Greek. Instruction may be given in English if foreign students attend the course.</td></tr>
<tr><td>IS THE COURSE OFFERED TO ERASMUS STUDENTS</td><td>Yes (in English)</td></tr>
<tr><td>COURSE WEBSITE (URL)</td><td>https://eclass.upatras.gr/courses/CEID1140/</td></tr>
</table>

(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*

After the successful completion of the course, the student:

- Will be able to describe the language of a deterministic, non-deterministic, multi-tape Turing machine.
- Will be able to design (in detail, using the transition diagram) Turing machines that decide or recognize simple languages.
- Will be able to describe (using high-level descriptions) Turing machines and algorithms for several decision problems.
- Will have developed intuition regarding the decidability or undecidability of languages and problems.
- Will have understood the definitions of basic undecidable languages (e.g., of the ones related to the halting problem) and will be able to use them in reduction-based proofs of undecidability results.

- Will have understood the notion of mapping reducibility and will be able to apply in undecidability and unrecognizability proofs.
- Will have understood the definition of complexity class p and will be able to identify problems in this class.
- Will have understood the definition of the complexity class NP and will be able to identify problems of this class.
- Will have developed intuition regarding the computational hardness of several languages and decision problems.
- Will have understood the basic NP-complete problems (satisfiability, finding cliques in graphs, vertex cover, Hamilton paths in graphs, etc.)
- Will be able to prove NP-completeness results through polynomial-time reductions.
- Will be able to prove statements regarding the structure of complexity classes P, NP, and coNP and their relationships.

In general, the students will have obtained a good knowledge of the basic principles that govern computation.

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

Search for, analysis and synthesis of data and information, with the use of the necessary technology
Decision-making
Working independently
Production of new research ideas
Production of free, creative and inductive thinking

(3) SYLLABUS

Turing machines, definitions and examples. Variations of the basic model. Multi-tape Turing machines, Non-deterministic Turing machines, Enumerators. Equivalence between different computation models. The definition of algorithm. Definition of Turing machine inputs.
Decidability. Decidable languages. The halting problem. The diagonalization method. Proof of undecidability of the halting problem. Reductions. Other undecidable languages. Undecidability proofs through reductions. Mapping reducibility. Computable functions. Definitions, basic theorems, examples. Proofs of language unrecognizability.
Measuring time complexity. Analysis of algorithms. Complexity relationships among different models. The class P. Polynomial-time deterministic Turing machines. Examples. The class NP. Non-deterministic Turing machines. Polynomia-time verifiers. Examples of problems of NP. The P versus NP question. Polynomial-time reductions, NP-completeness, definitions. The Cook-Leven Theorem. NP-completeness proofs. Satisfiability. Computing cliques in graphs. The vertex cover problem. The Hamiltonian path problem. The subset sum problem. Other NP-complete problems.

(4) TEACHING and LEARNING METHODS - EVALUATION

DELIVERY *Face-to-face, Distance learning, etc.*	**Face-to-face**

<table>
<tr><td>USE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY
Use of ICT in teaching, laboratory education, communication with students</td><td>Use of ICT in teaching (lectures in electronic form, Internet sources, etc.) and in communication with students (mailing list, course web site).</td></tr>
<tr><td>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.

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</td><td><table><tr><th>Activity</th><th>Semester workload</th></tr><tr><td>Lectures</td><td>26</td></tr><tr><td>Recitation sections</td><td>26</td></tr><tr><td>Independent study</td><td>60</td></tr><tr><td>Course total</td><td>112</td></tr></table></td></tr>
<tr><td>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.</td><td>Language of evaluation: Greek (English if needed, e.g., Erasmus+ students)

Final examination

The final examination is written, of graded difficulty, and can consist of multiple choice questions, questions for short answers, questions that require mathematical proofs or arguments as answers, problems and exercises</td></tr>
</table>

(5) ATTACHED BIBLIOGRAPHY

- Suggested bibliography:

- M. Sisper. Introduction to the Theory of Computation. 3rd Edition, Cencage Learning, 2013.
- H. Lewis & C. H. Papadimitriou. Elements of the Theory of Computation. Prentice-Hall, 1981.
- C. H. Papadimitriou. Computational Complexity. Addison-Wesley, 1994.
- J. E. Hopcroft, R. Motwani, & J. D. Ullman. Introduction to Automata Theory, Languages, and Computation. Addison-Wesley, 2001.

- Related academic journals:

As this is a course that introduces the student to very basic notions, which are covered in depth by the above-mentioned excellent textbooks, no academic journals are used.