COURSE OUTLINE

(1) GENERAL

SCHOOL	ENGINEERING		
ACADEMIC UNIT	DEPT. OF COMPUTER ENGINEERING AND INFORMATICS		
LEVEL OF STUDIES	UNDERGRADUATE		
COURSE CODE	CEID_NY301	SEMESTER	5th
COURSE TITLE	THEORY OF COMPUTATION		

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, Problem solving sessions	2(L), 2(PSS)	4
Add rows if necessary. The organisation of teaching and the teaching methods used are described in detail at (d).	TOTAL	4

COURSE TYPE *general background, special background, specialised general knowledge, skills development*	*special background*
PREREQUISITE COURSES:	
LANGUAGE OF INSTRUCTION and EXAMINATIONS:	Greek
IS THE COURSE OFFERED TO ERASMUS STUDENTS	Yes
COURSE WEBSITE (URL)	goo.gl/jzod59

(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*

Theory of Computation contains three central areas: automata, computability and complexity. These areas are linked by the following important question: What are the fundamental capabilities and inherent limitations of computers?

This question is interpreted differently in each of these three areas and corresponding answers are shaped accordingly. In the context of computability theory, the goal is to characterize the various problems as solvable or not. In complexity theory, the goal is to categorize solvable problems into easy and difficult and the central question is: What is it that makes some problems computationally hard and some other easy? One of the most important achievements of complexity theory is an elegant system of classifying problems according to their computational hardness.

This course essentially focuses on automata theory which deals with the definitions and properties of mathematical models of computation. These models are used in practice in several areas of computer science. For example, finite automata are used in string searching and pattern matching, word processing, compiler and hardware design. Another model, context-free grammars, is used in programming languages and artificial intelligence. Automata theory is a particularly interesting area that allows practice with formal definitions of computation which are highly significant in the theory

of computability and complexity where a precise definition of the computer is required.

Students who regularly participate in course activities and successfully complete the course:

- have knowledge and understanding for definitions and properties of mathematical models of computation (like, for instance, finite automata, push-down automata, Turing machine) as well as for corresponding methods (i.e., regular expressions, grammars, algorithms) for finite representation of particular classes of problems; students are therefore able to keep track of current developments at the cutting edge of their field of knowledge
- are able to use knowledge and understanding they have acquired in a way that shows a professional approach to their work or profession, and appropriately skilled to use mathematical models of computation for various problems within their field
- have the ability to collect and interpret relevant data (typically within their field) to form judgments that include reflection on relevant social, scientific or ethical issues
- are able to communicate information, ideas, problems and solutions to specialized and non-specialized audience
- have developed knowledge acquisition skills necessary to further continue their studies with a high degree of autonomy
- have become familiar with computational thinking and are able to exploit its advantages in scientific, professional and practical issues

In particular, students who regularly participate in course activities and successfully complete the course:

1. have knowledge of definitions and properties of mathematical models of computation and corresponding methods for finite representation of particular classes of problems
2. understand the relation between computational hardness of problems and relevant models of compuation
3. are able to develop and apply abstraction and modelling for algorithmic and compuational problems according to their hardness (I.e., resources required for solving them)
4. analyze problems / questions in order to gain understanding of their structure and components
5. suggest solutions to these problems guided by their computational hardness
6. evaluate findings (solutions or hardness results) through analysis
7. are familiar with computational thinking

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…*
	…….

Familiarity with computational thinking

Search for, analysis and synthesis of data and information, 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

Project planning and management

Respect for difference and multiculturalism

Showing social, professional and ethical responsibility and sensitivity to gender issues

Criticism and self-criticism

Production of free, creative and inductive thinking

(3) SYLLABUS

Theory of Computation contains three central areas: automata, computability and complexity. These areas are linked by the following important question: What are the fundamental capabilities and inherent limitations of computers?

This question is interpreted differently in each of these three areas and corresponding answers are shaped accordingly. In the context of computability theory, the goal is to characterize the various problems as solvable or not. In complexity theory, the goal is to categorize solvable problems into easy and difficult and the central question is: What is it that makes some problems computationally hard and some other easy? One of the most important achievements of complexity theory is an elegant system of classifying problems according to their computational hardness.

This course essentially focuses on automata theory which deals with the definitions and properties of mathematical models of computation. These models are used in practice in several areas of computer science. For example, finite automata are used in string searching and pattern matching, word processing, compiler and hardware design. Another model, context-free grammars, is used in programming languages and artificial intelligence. Automata theory is a particularly interesting area that allows practice with formal definitions of computation which are highly significant in the theory of computability and complexity where a precise definition of the computer is required.

Lectures proceed according to the following sections:

Set theory

- Proofs
- Sets, operations, alphabets, strings, languages, operations with languages

Regular languages

- Definition
- Model of computation: finite automata (DFA, NFA)
- Finite representation: regular expressions
- Equivalence of finite automata with regular expressions
- Closure properties of regular languages
- Non-regular languages

Context-free languages

- Model of computation: (non-deterministic) pushdown automata
- Finite representation: context-free grammars
- Equivalence of (non-deterministic) pushdown automata with context-free grammars
- Closure properties of context-free languages
- Non-context-free languages

Church-Turing thesis

- Model of computation: Turing machine
- Finite representation: algorithms

(4) TEACHING and LEARNING METHODS - EVALUATION

<table>
<tr><td>DELIVERY
Face-to-face, Distance learning, etc.</td><td>Face to face, Distance learning</td></tr>
<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 (online lectures, course website, extensive use of Web resources), in communication/collaboration with students (mailing lists, social networks (Facebook), course website, Doodles) and in the process of progress monitoring and evaluation (use of specialized software for the monitoring and evaluation of student progress)</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>Problem solving sessions</td><td>26</td></tr>
<tr><td>Intense cooperation among professor and students also using ICT</td><td>6</td></tr>
<tr><td>Independent study</td><td>62</td></tr>
<tr><td></td><td></td></tr>
<tr><td>Course total
(25-30 hours per credit)</td><td>120</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>Written examination</td></tr>
</table>

(5) ATTACHED BIBLIOGRAPHY

- Suggested bibliography:

INTRODUCTION TO THE THEORY OF COMPUTATION, Michael Sipser
ELEMENTS OF THE THEORY OF COMPUTATION, Harry R. Lewis, Χρίστος Χ. Παπαδημητρίου

- Related academic journals:

Theoretical Computer Science, Elsevier

Theory of Computing Systems, Springer