INSTITUTION	NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS						
SCHOOL	SCHOOL OF SCIENCE						
DEPARTMENT	INFORMATICS AND TELECOMMUNICATIONS						
COURSE LEVEL	UNDERGRADUATE						
COURSE TITLE	**Compilers**						
COURSE CODE	**K31**		**Semester**	**6**	**ECTS**	**6**	
TEACHING HOURS per week	**THEORY**	**4**	**SEMINAR.**	**1**	**LABORATORY**	**0**	
COURSE TYPE	**Select one of the following and delete the rest** Track Compulsory (EYM) **K**: A; **E1**: ; **E2**: ; **E3**: Y; **E4**: B; **E5**: ; **E6**: *Fill the table as in the curriculum:* Track (A-Computer Science, B- Computer Engineering) / Specialization Compulsory (Y) / Core Specialization (B)/ Elective Specialization (E)						
URL	https://eclass.uoa.gr/courses/D38/						
EXPECTED PRIOR KNOWLEDGE/ PREREQUISITES AND PREPARATION:	K08 Data Structures K10 Object-Oriented Programming Recommended K14, Computer Architecture I						
TEACHING AND EXAMINATIONS LANGUAGE:	GREEK						
THE COURSE IS OFFERED TO ERASMUS STUDENTS	YES						

COURSE CONTENT
Explore the fundamental concepts and techniques behind a compiler: (1) Formal languages: regular languages, context-free languages, attribute grammars; (2) Meta-tools to create lexical analyzers; (3) Parsing: top-down and bottom-up, error recovery, meta-tools to use and create syntax analyzers; (4) Symbol tables. Semantic analysis: kinds of semantic checking, static type systems, dynamic type checking; (5) Generation of intermediate code; (6) Optimization, register allocation; (7) Generation of object code.

STUDENT LEARNING OBJECTIVES

Teaching-Learning Goals
Learning of key concepts in compilers and language implementation, develop programming abilities

Expected Learning Outcomes
Upon successful completion of the course the student will be able to:

- design a simple formal grammar
- convert an input-output specification into a finite automaton
- process an automaton algorithmically
- implement a simple programming language
- specify the behavior of compilers over complex programs
- design and implement a single language processor or program analyzer

TEACHING AND LEARNING METHODS - ASSESSMENT	
TEACHING METHOD	In Class (Face to Face)
USE OF INFORMATION AND COMMUNICATION TECHNOLOGIES	Learning process supported by web page, providing homeworks and tutorials Interaction and support forum on the Piazza platform Live transmission of lectures Ability to track recorded lectures
TEACHING ORGANIZATION *Describe in detail the way and methods of teaching:* *Enhanced Lectures,* *Online Lectures,* *Seminars,* *Tutorial,* *Laboratory,* *Laboratory Exercise,* *Study & analysis of literature,* *Practice (Positioning),* *Interactive teaching,* *Developing a project,* *Individual / group work* *Telework (reference to tools) etc.* *Details of the student's study hours for each learning activity and hours of non-guided study are shown to ensure that the total workload at the semester corresponds to the ECTS*	

Activity	***Student Workload*** *(hours)*
Lectures	52
Tutorial	13
Laboratory	0
Teamwork in a case study	0
Small individual exercises	40
Independent Study	45
Total Course (25 hours of workload per unit of credit)	***150***

ASSESSMENT OF STUDENTS

Description of the assessment process

Assessment Methods, Formative or Concluding, Multiple Choice Test, Quick Response Questions, Test Development Questions, Problem Solving, Written Work, Report / Report, Oral Examination, Public Presentation, Laboratory Work, Other / Other

Fully defined evaluation criteria are mentioned and if and where they are accessible to students.

Describe explicitly methods, evaluation tools and provided feedback.
The table below is supplemented accordingly.

Assessment methods	***Number***	***Percentage***
Written examination	1	50%
Exercises	2	50%

LITERATURE AND STUDY MATERIALS / READING LIST

Alfred V. Aho, Monica S. Lam, Ravi Sethi and Jeffrey D. Ullman Compilers: Principles, Techniques, and Tools. 2nd edition. Addison-Wesley, 2007. http://dragonbook.stanford.edu/

Nikolaos S. Papaspyrou and Emmanuel St. Skordalakis, Compilers, Symmetria, Athens, 2002.

K. Lazos, P. Katsaros, Z. Karaiskos, Compilers of Programming Languages: Theory and Practice, Thesaloniki 2004 http://delab.csd.auth.gr/~katsaros/CompilersBook.htm