INSTITUTION	NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS						
SCHOOL	SCHOOL OF SCIENCE						
DEPARTMENT	INFORMATICS AND TELECOMMUNICATIONS						
COURSE LEVEL	UNDERGRADUATE						
COURSE TITLE	**Software Engineering**						
COURSE CODE	**ΥΣ09**		**Semester**	**8**	**ECTS**	**6**	
TEACHING HOURS per week	**THEORY**	**3**	**SEMINAR.**	**1**	**LABORATORY**		
COURSE TYPE	**Select one of the following and delete the rest** Electives (ΠΜ) **K** A; **E1**; **E2**; **E3** B; **E4**; **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/DI423/						
EXPECTED PRIOR KNOWLEDGE/ PREREQUISITES AND PREPARATION:	K10 – Object Oriented Programming						
TEACHING AND EXAMINATIONS LANGUAGE:	GREEK						
THE COURSE IS OFFERED TO ERASMUS STUDENTS	NO						

COURSE CONTENT
The course offers an introduction to the field of Software Engineering, highlighting the various software design, quality and management issues that occur during the development of large software projects, while presenting the current state of the art in addressing these issues. Specifically, the course covers software life-cycle phases and their management methodologies, focusing on (a) the agile methodology in general and Scrum in particular, (b) the DevOps methodology and specifically the issues underlying version control, build automation and test automation. The course also deals with software design and architecture, discussing analysis and modeling of requirements, object-oriented design and architectural patterns, emphasizing the architectural choices that arise from the CAP theorem. Finally, the course also presents the latest techniques used in the development of Web-based user interfaces.

STUDENT LEARNING OBJECTIVES
Teaching-Learning Goals-Expected Learning Outcomes Upon successful completion of the course the student will be able to: • Explain the software life-cycle, its phases and the methodologies used to manage them. • Describe, classify, model and analyze software requirements. • Classify, compare and/or select the basic design and architectural patterns that build a complex software application. • Design, specify and develop large software using best practices. • Organize, manage and evaluate the tasks required to build a large software application.

TEACHING AND LEARNING METHODS - ASSESSMENT	
TEACHING METHOD	In Class (Face to Face)
USE OF INFORMATION AND COMMUNICATION TECHNOLOGIES	Learning process supported by the e-class platform (Units, Announcements, Documents, Texts, Student teams, Discussions). Email communication. Live transmission of lectures. Ability to track recorded lectures. Utilization of software code repositories in the course's laboratory and team projects.
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
Laboratory	13
Team project (5-7 students)	60
Individual study	25
Total	***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.*	*The written examination contains multiple choice tests, quick response questions and a test development – problem solving question. The team project involves group work with intermediary milestones that receive direct feedback from the instructor as well as a public presentation at its completion.*

Assessment methods	***Number***	***Percentage***
Written examination	1	50%
Team project	2	50%

LITERATURE AND STUDY MATERIALS / READING LIST

- Software Engineering: Theory and Practice, 2nd Edition, PFLEEGER, Kleidarithmos publishing (in Greek)
- Software Engineering, 8th Edition, SOMMERVILLE, Kleidarithmos publishing (in Greek)
- Software Engineering, 8th Edition, PRESSMAN, Tziola publishing (in Greek)

The course also offers an extended reading list available at its website.