MODULE OUTLINE

1. GENERAL INFORMATION

SCHOOL	SCHOOL OF SCIENCE AND TECHNOLOGY		
PROGRAM COURSE	INFORMATICS		
LEVEL OF STUDY	UNDERGRADUATE		
MODULE CODE	PLI-24	**YEAR OF STUDY**	3rd
MODULE TITLE	Software Design		
INDEPENDENT TEACHING ACTIVITIES *in case credits are awarded for separate components/parts of the course, e.g. in lectures, laboratory exercises, etc. If credits are awarded for the entire course, give the weekly teaching hours and the total credits*		**HOURS**	**CREDIS**
Weekly teaching hours * 32 weeks		14-16	18 ECTS
COURSE TYPE *Background knowledge, Scientific expertise, General Knowledge, Skills Development*	Scientific expertise, Skills Development Compulsory		
PREREQUISITE MODULES:	No		
LANGUAGE OF INSTRUCTION AND EXAMS	GREEK		
THE MODULE IS OFFERED TO ERASMUS STUDENTS	No (due to annual duration of the module)		
MODULE WEBSITE (URL)	https://www.eap.gr/education/undergraduate/computer-science/topics/#sxediasmos_logismikoi Each module has its own space in the Learning Management System of EAP (http://study.eap.gr), with controlled access (use of code) for students and teaching staff.		

2. LEARNING OUTCOMES

Learning Outcomes

- *The course learning outcomes, specific knowledge, skills and competences of an appropriate (certain) level, which students will acquire upon successful completion of the course, are described in detail. It is necessary to consult:*

Part A: Compilers

Upon successful completion of the course, the student will be able:

- To explain what a compiler and what an interpreter is and describe the translation stages.
- To use regular expressions and automata for lexical analysis.
- To use context-free grammars and parse trees for syntactic analysis.
- To apply top-down and bottom-up techniques for syntactic analysis.

- To apply grammar transformation techniques to make a grammar suitable for recursive-descend parsing and LL(1) detection.
- To implement a lexical and syntax analyzer using the Java programming language.
- To design and use a symbol table.
- To explain and construct syntactically directed translation schemes.
- To escribe intermediate code representations and optimization mechanisms.
- To describe principles of target code generation.

Part B: Software Engineering II Object Oriented Analysis and Design

Upon successful completion of the course, the student will be able:

- To describe and explain the basic concepts of object-oriented software technology (software, tools, processes, methodologies, life cycle, etc.).
- To know ICONIX object-oriented development methodology and, secondarily, Unified Process, as well as the phases, iterations, and activities it involves.
- To use ICONIX methodology for the development of an object-oriented software system.
- To develop analysis models by using UML.
- To develop the use case model and the corresponding diagrams.
- To develop the problem domain model.
- To design a software system
- To develop various UML diagrams such as: robustness diagrams, sequence diagrams, class diagrams, state diagrams, software components diagrams, software packages diagrams, and deployment diagrams.
- To develop the software using an object-oriented programming language.
- To conduct quality control (coupling, cohesion, etc.) of the software system.
- To prepare the necessary documentation artifacts.

Part C: Programming Languages II – Object Oriented Programming

Upon successful completion of the course, the student will be able:

- To outline and describe the basic object-oriented programming concepts such as class, object, instance, inheritance relationship, aggregation relationship, etc.
- To describe the proper syntax of java data structures, types and operators.
- To implement classes, instances, methods and class relationships.
- To implement the principles data hiding, overriding and polymorphism.
- To write small and medium scale object-oriented applications.
- To read data files and use data bases for data storage.
- To implement exception handling mechanisms.
- To write multi-threaded applications exception handling mechanisms
- To create graphical user interfaces for applications using java libraries.

Additional material: Software Project Panning -Object oriented methodologies

Upon successful completion of the course, the student will be able:

- To describe and explain key concepts of project management (project, project management phases, organizational structures, project scope, scheduling, etc.).
- To understand the different structures used for organizing projects and teams.
- To create project plans including Gantt charts.

- To perform project network analysis using CPM and PERT project management techniques.
- To estimate the required project effort using the use case points.
- To manage resources and perform resource leveling for a project.
- To describe and apply cost estimation techniques for a project.
- To describe and explain the basic principles of software quality assurance and use quality metrics to measure software quality.
- To describe the concept of software risk in software development projects, and predict-manage risks that are likely to occur in a project.

General Competences

Taking into consideration the general competences that students/graduates must acquire (as those are described in the Diploma Supplement and are mentioned below), at which of the following does the course attendance aim?

Search for, analysis and synthesis of data and information by the use of appropriate technologies,
Adapting to new situations
Decision-making
Individual/Independent work
Group/Team work
Working in an international environment
Working in an interdisciplinary environment
Introduction of innovative research
Project planning and management
Respect for diversity and multiculturalism
Environmental awareness
Social, professional and ethical responsibility and sensitivity to gender issues
Critical thinking
Development of free, creative and inductive thinking
……
(Other…….citizenship, spiritual freedom, social awareness, altruism etc.) …….

Search for, analysis and synthesis of data and information by the use of appropriate technologies
Adapting to new situations
Decision-making
Individual/Independent work
Group/Team work
Project planning and management
Social, professional and ethical responsibility and sensitivity to gender issues
Critical thinking
Development of free, creative and inductive thinking

3. MODULE CONTENT

The purpose of the module is to introduce to students with the concepts, tools, and processes related to “Software Design”. The main objectives of the module are, students:

- to get acquainted with concepts related to compilers and interpreters,
- to be able to build basic parts of a compiler,
- to learn with object-oriented technology and software analysis,
- to familiarise to the basic concepts of software development tools,
- to be introduced to basic software and software quality assurance and, finally
- to learn object-oriented programming through Java programming language.

The key subjects of the module are:

- Compilers

- Software engineering II
- Programming languages II – Object oriented programming

4. TEACHING METHODS--ASSESSMENT

<table>
<tr><td>MODES OF DELIVERY
Face-to-face, in-class lecturing, distance teaching and distance learning etc.</td><td>Distance education with five Group Counseling Meetings (OSS) during the academic year on weekends.</td></tr>
<tr><td>USE OF INFORMATION AND COMMUNICATION TECHNOLOGY
Use of ICT in teaching, Laboratory Education, Communication with students</td><td>Remote meetings tools (skype for business),
Presentation software (e.g. power point),
Specialized software in the subjects under study:
- Netbeans (https://netbeans.org/) for the java programming language
- Visual Paradigm for UML (https://www.visual-paradigm.com/)
- Antlr (https://www.antlr.org/) for compilers.
- Trello or similar tools for project team collaboration

Additionally, the students use office automation tools, web browsers and e-readers for digital books.</td></tr>
<tr><td>MODULE DESIGN
Description of teaching techniques, practices and methods: Lectures, seminars, laboratory practice, fieldwork, study and analysis of bibliography, tutorials, Internship, Art Workshop, Interactive teaching, Educational visits, projects, Essay writing, Artistic creativity, etc

The study hours for each learning activity as well as the hours of selfdirected study are given following the principles of the ECTS.</td><td>
<table>
<tr><th>Activity</th><th>Annual Workload</th></tr>
<tr><td>5 OSS (* 4 hours)</td><td>20</td></tr>
<tr><td>Problem Solving</td><td>15</td></tr>
<tr><td>Preparation of Assignments (4 assignments * 16 hours)</td><td>64</td></tr>
<tr><td>Group Assignment</td><td>20</td></tr>
<tr><td>Exams</td><td>3.5</td></tr>
<tr><td>Individual study</td><td>325,5-389,5</td></tr>
<tr><td>Total module workload (hours)</td><td>448-512</td></tr>
<tr><td></td><td></td></tr>
</table>
</td></tr>
<tr><td>STUDENT PERFORMANCE EVALUATION/ASSESSMENT METHODS
Detailed description of the evaluation procedures.</td><td>Elaboration of written assignments during the academic year, the average of the grades of which participates in the formation of the final grade of module by 30%, if there is a passable in the final or repetitive examinations. In the final written exams the</td></tr>
</table>

Language of evaluation, assessment methods, formative or summative (conclusive), multiple choice tests, short- answer questions, open-ended questions, problem solving, written work, essay/report, oral exam, presentation, laboratory work, other……etc. *Specifically defined evaluation criteria are stated, as well as if and where they are accessible by the students*	grade of the written assignments participates in the formation of the final grade of module by 70%. All the criteria are posted, both in each written assignment (in the LMS study.eap.gr), as well as in the general regulation of HOU at: https://www.eap.gr/education/study-regulations/ An optional examination is also performed in this specific module.

(6) SUGGESTED BIBLIOGRAPHY

- *Suggested bibliography:*

HOU Publications:

- Volume A: Compilers, HOU, Patras 2008. PLH2/1/09
- Volume B: Software Engineering II, HOU, Patras 2008. PLH24 /2/09
- Volume C: Programming Languages II (Object Oriented Programming), Patras 2001. PLH24/3
- Volume D: IT Project Planning-Object Oriented Methodologies, HOU, Patras 2008. PLI24/4/09

Third party books

- Aho A., Lam M.,, Sethi R., and Ullman J. (2014). Compilers: principles, techniques and tools, New Technologies Publications, 3rd Edition.
- Deitel, P., and Deitel, H. (2015). Java Programming. M. Giourdas Publications, 10th Edition

-Related scientific Journals:

1) Java magazine, ORACLE (https://blogs.oracle.com/javamagazine/)
2) IEEE Software, IEEE (https://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=52)
3) IEEE Spectrum, IEEE (https://spectrum.ieee.org/)
4) Communications of the ACM, ACM (https://cacm.acm.org/)