COURSE OUTLINE

(1) GENERAL

SCHOOL	ENGINEERING		
ACADEMIC UNIT	INFORMATICS AND COMPUTER ENGINEERING		
LEVEL OF STUDIES	UNDERGRADUATE		
COURSE CODE		**SEMESTER**	**2nd**
COURSE TITLE	OBJECT-ORIENTED PROGRAMMING		

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	3	
Tutoring	2	
Laboratory exercises	1	
Add rows if necessary. The organisation of teaching and the teaching methods used are described in detail at (d).	6	6

COURSE TYPE general background, special background, specialised general knowledge, skills development	Background, Skills Development
PREREQUISITE COURSES:	Computer programming
LANGUAGE OF INSTRUCTION and EXAMINATIONS:	Greek
IS THE COURSE OFFERED TO ERASMUS STUDENTS	Yes (English)
COURSE WEBSITE (URL)	

(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

The aim of the course is to acquire a foundation in object-oriented design and software development using the C++ implementation language.
Upon completion of the course, students will be able to:

- Explain the basic concepts of object-oriented programming and the how they are implemented in the C++ language
- Read and modify well some C++ programs
- Make correct object-oriented design choices for small and medium sized software systems
- Implement programming solutions that incorporate features inheritance and polymorphism features.
- Implement sound, modular, reusable and maintainable code
- Develop programming solutions using standardized linguistic constructs of C++, but also features such as type conversions, type definitions , exceptions and function/class patterns

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 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 Respect for the natural environment Showing social, professional and ethical responsibility and sensitivity to gender issues Criticism and self-criticism Production of free, creative and inductive thinking …… Others… …….

- Autonomous Work
- Teamwork
- Search, analysis and synthesis of data and information using the necessary technologies
- Exercising criticism and self-criticism through peer evaluation exercises

(3) SYLLABUS

- Principles and mechanisms of object-oriented analysis and software design.
- Overview of object-oriented programming in the C++ language.
- Basics of C++, differences with C.
- Organization and compilation of C++ programs.
- Definition of new types. Abstraction in data.
- Classes I (build - destroy functions, build functions copy construction, inline functions, overloading).
- Namespaces. Inheritance.
- Classes II (virtual and fully virtual functions, abstract classes).
- Function templates and class templates.
- The C++ Standard Template Library (STL).
- Input / Output in C++.
- Exception handling.

(4) TEACHING and LEARNING METHODS - EVALUATION

DELIVERY Face-to-face, Distance learning, etc.	• Face to face
USE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY Use of ICT in teaching, laboratory education, communication with students	• Integrated software development environments. • Supporting the learning process through • electronic platform (e-class) of the University.
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	(see table below)
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.	A. Written final examination (60%) including: • Short answer questions • Design and development of object-oriented software B. Laboratory exercises (20%) C. Project presentation (20%) For successful completion, a grade of at least 5/10 in the Written Final Examination is necessary

Activity	Semester workload
Lectures	39
Tutoring	26
Laboratory exercises	13
Preparation of a study (project) integrated analysis, design, development object-oriented software	25
Autonomous Study	47
Course total (25 hours load per credit hour unit)	150

(5) ATTACHED BIBLIOGRAPHY

- Suggested bibliography:
1.Cleo Sguroopoulou, "Object-Oriented Programming with C++", Tsotras, 2019
2. Edition), "Keydarithmos", 2014
3. Bruce Eckel, "Thinking In C++" (Second Edition), Volume One & Two, Prentice Hall, 2000
ISO International Standard: Programming Languages - C++, 1998
Stanley B. Lippman, Josee Lajoie, "C++ Primer" (4th Edition), Addison-Wesley, 2005
Herbert Schildt, "C++: The Complete Reference" (4th Edition), McGraw-Hill, 2004
The Standard Template Library: http://www.sgi.com/tech/stl
8. Bertrand Meyer, "Object-Oriented Software Construction" (2nd Edition)