COURSE OUTLINE

(1) GENERAL

SCHOOL	School of Engineering		
ACADEMIC UNIT	Department of Naval Architecture		
LEVEL OF STUDIES	Undergraduate		
COURSE CODE	NAOME1105	**SEMESTER**	1st
COURSE TITLE	**INTRODUCTION TO COMPUTER PROGRAMMING**		

INDEPENDENT TEACHING ACTIVITIES	**WEEKLY TEACHING HOURS**	**CREDITS (ECTS)**
Lectures	3	4

COURSE TYPE *general background, specialbackground, specialised general knowledge, skills development*	General background
PREREQUISITE COURSES:	
LANGUAGE OF INSTRUCTION and EXAMINATIONS:	Greek
IS THE COURSE OFFERED TO ERASMUS STUDENTS	Yes
COURSE WEBSITE (URL)	https://eclass.uniwa.gr/courses/NA188/

(2) COURSE GOALS / LEARNING OUTCOMES

The focus of this course is the introduction to contemporary computer systems and modern programming languages. This course places emphasis on the development of algorithmic techniques to demonstrate the methodological problem solving approach in a variety of fields. A basic goal of the course is to familiarize students with modern integrated programming environments and the development of programs for mathematical computations and visualization of the results. The python programming language and its modules are used throughout the course.
After completing this course the student shall be able to:

- Comprehend and describe the basic functionality of the architectural components of a computer system.
- Understand how data and information is organized and represented within a computer system
- Use the basic data and algorithmic structures available in modern computer languages
- Analyze a problem in it's primary components and develop an algorithmic solution for such problems
- Understand basic algorithm representation and encoding techniques
- Analyze a problem and structure an algorithmic solution
- Utilize modern integrated programming environments for Python with emphasis on Jupyter Notebooks and Jupyter Lab
- Develop programs in the Python programming language with the use of good programming practices and programming techniques
- Utilize tools and methodologies for program debugging
- Understand and deploy the basic principles of procedural and vector programming

- Use current data types like tuples, sets, sequences, dictionaries and lists to develop programs
- Develop programs that perform scientific computations that include scalars, vectors and matrices
- Provide visualizations through two-dimensional and three-dimensional graphs
- Develop and modify Python programs and functions

(3) COURSE CONTENT / SYLLABUS

- Computer architecture and components
- Hardware – Software
- Principles of Computer Programs. Introduction to computer languages
- Problem solving methodologies, Design principles of computer programs, Introduction to algorithms, Flow diagrams, Pseudocode
- Integrated development environments
- Introduction to Python. Online development environments like Jupyter Labs and the use of Notebooks
- Variables and expression. Logical expressions, Input and output functions
- Basic data types (arithmetic, logical, strings, records) and computations between different data types
- Flow control structures, Loops and Functions
- Contemporary data types (lists, tuples, sets, sequences and dictionaries)
- Matrices and use of Python modules (NumPy, SciPy). Matrix computations and addressing
- Mathematical functions and numerical applications
- File input and output
- Program debugging
- Graphs through the use of Matplotlib

(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	• Use of ICT methodologies in teaching. • Learners support through email, and the asynchronous electronic platform (e-class) • Lectures available on the course webpage (e-class)
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*	

Activity	*Workload (hours)*
Lectures	26
Exercises	13
Homework assignments (problem solving with code development in Python programming language)	26
Study of Lectures	52

principles of the ECTS		
	Course total	***117***
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*	i) Written final examination (60%). ii) Problem solving / code development in Python (40%).	

(5) ATTACHED BIBLIOGRAPHY

1. Καρολίδης Δ.Α., 2016, Μαθαίνετε εύκολα Python, Εκδόσεις Άβακας.
2. Gaddis, T., 2014, Ξεκινώντας με την Python, Εκδόσεις DaVinci.
3. Αβούρης Ν. κ.α., 2018, Python - Εισαγωγή στους υπολογιστές
4. Schneider D., 2016, Εισαγωγή στον Προγραμματισμό με την Python, Εκδόσεις Γκιούρδας
5. Μανής, Γ., 2015. Εισαγωγή στον Προγραμματισμό με αρωγό τη γλώσσα Python. [ηλεκτρ. βιβλ.] Αθήνα:Σύνδεσμος Ελληνικών Ακαδημαϊκών Βιβλιοθηκών. Διαθέσιμο στο: http://hdl.handle.net/11419/2745