PRINCIPLES OF PROGRAMMING LANGUAGES AND COMPILERS

COURSE OUTLINE

Responsible: Nikolaos Spanoudakis

1. GENERAL			
SCHOOL	School of Engineering		
ACADEMIC UNIT	Department of Electronic Engineering		
LEVEL OF STUDIES	Undergraduate		
COURSE CODE	0806.8.012.0	**SEMESTER**	2nd
COURSE TITLE	Principles of Programming Languages and Compilers		

INDEPENDENT TEACHING ACTIVITIES if credits are awarded for separate components of the course	**WEEKLY TEACHING HOURS**	**CREDITS**
	6	5
Total	**6**	**5**

COURSE TYPE general background, special background, specialised general knowledge, skills development	Specialization, Scientific Area
PREREQUISITE COURSES	None
LANGUAGE OF INSTRUCTION and EXAMINATIONS	English
OFFERED TO ERASMUS STUDENTS	Yes (in English) — Spring Semester
COURSE WEBSITE (URL)	https://eclass.hmu.gr/courses/EE342/

2. LEARNING OUTCOMES

Learning outcomes

Upon completion of the course, students will be able to:

1. List the different programming language paradigms and their main characteristics, including functional and logic programming.
2. List the main criteria for good design and success of a programming language.
3. Explain the operation of compilers and interpreters, their differences and the environments in which they are used.
4. Construct formal grammars, which are part of formal languages, the theoretical model for describing the syntax of programming languages.
5. Implement lexical analyzers, both through their theoretical description with finite automata and regular expressions and by programming with related regex tools.
6. Implement parsers, both through their theoretical description with context-free grammars using BNF notation and by programming with related tools such as xtext.
7. Explain how the various programming languages implement and handle higher-level syntactic elements such as variables, expressions and statements.
8. Define appropriate data types in their programs, exploiting the capabilities offered by different programming languages.
9. Explain the two basic scoping rules (static and dynamic) and the ways they are implemented in different programming languages.

10. List the memory allocation and management mechanisms during program execution.
11. Explain the types, components, characteristics and design challenges of using subprograms.
12. Program using subprograms, exploiting the different parameter-passing methods supported by programming languages.

General Competences

Search for, analysis and synthesis of data and information, using the necessary technologies

Autonomous work

Written communication

Time management

Adapting to new situations

Decision making

Promotion of free, creative and inductive thinking

Computer use

3. SYLLABUS

1. Introduction to programming languages
2. Historical evolution of programming languages
3. Compilers and interpreters
4. Syntax of programming languages
5. Lexical analysis
6. Syntax analysis (parsing)
7. Variables, Expressions, Statements
8. Data Types
9. Scope and memory binding time
10. Subprograms
11. Functional programming
12. Logic programming

4. TEACHING and LEARNING METHODS - EVALUATION

DELIVERY Face-to-face, Distance learning, etc.	Face-to-face theoretical and practical teaching with laboratory exercises
USE OF INFORMATION AND COMMUNICATIONS TECHNOLOGY Use of ICT in teaching, laboratory education, communication with students	The lecture slides and additional supplementary support material are freely available on the course website. Communication with students takes place through a dedicated online forum.

TEACHING METHODS
The manner and methods of teaching are described in detail.

Activity	Semester workload
Lectures	26
Laboratory exercises	26
Tutorial Exercises	19
Independent study, preparation and exercises	50
Examinations	3
Tutorials	26
Course total	**150**

STUDENT PERFORMANCE EVALUATION
Description of the evaluation procedure

I. Written final exam (FE) (40-60%)
- Problem solving
- Multiple-choice questions
- Short-answer questions
- Comparative evaluation of theory elements

II. Laboratory assignments (LA) (20%)
- Oral examination
- Grading of the submitted assignment

III. Written midterm test (MT) (20%)
- Problem solving
- Multiple-choice questions
- Short-answer questions
- Comparative evaluation of theory elements

IV. Written Assignment (WA) (20%)
- Grading of the submitted assignment

The course grade (FE*0.4 + LA*0.2 + MT*0.2 + WA*0.2) must be at least five (5).
Students may have at most one absence from the laboratory exercises.
The MT is optional (if it is not taken, its weight is transferred to the FE).
The assessment criteria are accessible to students on the course website and are announced in the first lecture.

5. ATTACHED BIBLIOGRAPHY

- Keith D. Cooper, Linda Torczon. Engineering a Compiler (Greek translation). Crete University Press, 2018
- Panagiotis Katsaros, Georgios Makris, Anastasios Temperekidis. Programming Languages and Compilers. Markou & Co., 2020 (in Greek)
- Katerina Georgouli. Compilers. Kallipos Open Academic Editions, 2022 (in Greek)
- Georgios Manis. Compiler Design and Development Handbook. Kallipos Open Academic Editions, 2023 (in Greek)

Foreign-language

- Keith D. Cooper and Linda Torczon. Engineering a Compiler. 3rd edition, Morgan Kaufmann, 2022
- The Elements of Computing Systems, second edition: Building a Modern Computer from First Principles. The MIT Press, 2021