COURSE DATA

DATA SUBJECT

Code: 34672
Name: Programming languages
Cycle: Undergraduate Studies
ECTS Credits: 6
Academic year: 2025-26

STUDY (S)

Degree	Center	Acad. year	Period
1400 - Degree in Computer Engineering	Escola Tècnica Superior d'Enginyeria	3	First quarter
1407 - Degree in Multimedia Engineering	Escola Tècnica Superior d'Enginyeria	4	First quarter

SUBJECT-MATTER

Degree	Subject-matter	Character
1400 - Degree in Computer Engineering	Computing and programming	COMPULSORY
1407 - Degree in Multimedia Engineering	Optatividad	ELECTIVES

COORDINATION

ADSUARA FUSTER JOSE ENRIQUE

AREVALILLO HERRAEZ MIGUEL

SUMMARY

"Programming Languages" is a 6 ECTS credit compulsory subject in the Third Year of the Bachelor degree in Computer Science. The subject is part of the module "Programming and Computation"

The content builds on the programming knowledge previously acquired and elaborates on advanced aspects. In particular, the following contents are covered: the compilation process and keys aspects of functional programming.

An integrative approach is adopted. Building on contents learned in the subject "Automata, Formal Languages and Applications", the subject starts introducing the compilation process, by using generation tools to construct lexical analyzers and parsers. In addition, the functional programming paradigm is presented. This paradigm is studied from the perspective of the application of concepts such as

immutability and higher-order functions in writing code.

PREVIOUS KNOWLEDGE

RELATIONSHIP TO OTHER SUBJECTS OF THE SAME DEGREE

There are no specified enrollment restrictions with other subjects of the curriculum.

OTHER REQUIREMENTS

To take this course, it is recommended to master the basic concepts of programming and have passed the subjects in the same area such as Data Structures and Algorithms, and Automata, Formal Languages, and Applications; the subjects of Computer Organization; and the subject area of Computer Science. In particular, the content on grammars and formal languages studied in Automata, Formal Languages, and Applications is essential for understanding the concepts related to the compilation process.

COMPETENCES / LEARNING OUTCOMES

1400 - Degree in Computer Engineering

G4 - Ability to define, evaluate and select hardware and software platforms for the development and implementation of computer systems, services and applications, in accordance with both the knowledge and the specific skills acquired in the degree.

G5 - Ability to design, develop and maintain computer systems, services and applications using software engineering methods as an instrument for quality assurance, in accordance with both the knowledge and the specific skills acquired in the degree.

G9 - Ability to solve problems with initiative, decision making, autonomy and creativity. Ability to communicate and transmit the knowledge, skills and abilities of a computer engineer.

R6 - Knowledge and application of basic algorithmic procedures of computer technology to design solutions to problems, by analysing the suitability and complexity of the algorithms proposed.

R7 - Knowledge, design and efficient use of the types and structures of data most suitable for solving a problem.

R8 - Ability to analyse, design, build and maintain applications in a robust, secure and efficient manner by choosing the most suitable paradigm and programming languages.

DESCRIPTION OF CONTENTS

1. Introduction.

Historical evolution of programming Languages.
Main characteristics of programming languajes.
Programming Language classification.

2. Language Processors.

Types: compilers, interpreters.
Sintax: criteria, syntatic elements.
Source code analysis:
- Lexical analysis.
- Syntactic analysis (descending and ascending).
- Semantic Analysis.

Code Generation:
- Intermediate code.
- Optimization.

Tools.

3. Functional Programming.

Why functional programming.
What is functional programming.
Application of functional paradigm concepts.

WORKLOAD

PRESENCIAL ACTIVITIES

Activity	Hours
Theory	30,00
Laboratory	20,00
Classroom practices	10,00
Total hours	**60,00**

NON PRESENCIAL ACTIVITIES

Activity	Hours
Attendance at other activities	0,00
Individual or group project	0,00
Independent study and work	60,00
Preparation of lessons	20,00
Preparation for assessment activities	10,00

Resolution of case studies	0,00
Total hours	**90,00**

TEACHING METHODOLOGY

In-class learning activities in this subject include both theoretical and practical ones. Theoretical activities, in the form of lectures, aim at developing an integrative view of the contents. This type of lessons cover the key and most complex aspects of the subject and attempt to foster student participation. The knowledge acquired during the lecture sessions is consolidated by a series of practical activities. These include:

• Problem sessions in the classroom

• Tutorial sessions to discuss independent work

• Laboratories

• In class quizzes

In addition to in class activities, a reasonable amount of independent work is required, to prepare for lectures, search for contents, solve a wide range of problems, complete practical course works and study for the quizzes. The University Learning Management System (LMS) Aula Virtual will be used to support the delivery of this course. This LMS will be used to disseminate materials and to facilitate course work submissions

EVALUATION

In this course, the evaluation will be carried out through continuous assessment, which will include three written tests covering theoretical, practical, and problem content (P1, P2, and P3). These tests will generally be conducted during the first half of the semester (P1), during the second half of the semester (P2), and outside class hours in the exam period (P3). Additionally, the practical work will be evaluated based on a project (Pr). The tests will not eliminate any material and will therefore have increasing weight in the final grade. To pass the course, it will be necessary to obtain at least a grade of 5 in the exam part computed as Pe = 0.15P1 + 0.35P2 + 0.5P3 and a grade of at least 4 in the course project (Pr). The final grade (C) will be obtained with C = 0.7Pe + 0.3*Pr.

In the second call, the grade will be the one obtained in the test taken in the corresponding exam period. The obligation to submit the final project remains (which may be resubmitted until the day of the exam), requiring a minimum grade of 5.

All tests may contain sections or parts that are eliminatory.

In all cases (first and second call), the student must submit all practical work and complementary activities correctly completed to pass the course, within the deadline and in the proper form. Otherwise, the

maximum obtainable grade will be 4.

The use of automatic code generation tools or answers based on language models / artificial intelligence tools should be explicitly declared and justified, and is expressly prohibited in the exams. Non-declared used of such tools would be considered a case of plagiarism.

In any case, the evaluation of this subject will be done in compliance with the University Regulations in this regard, approved by the Governing Council on 30th May 2017 (ACGUV 108/2017). Copying or plagiarism of any activity that is part of the evaluation will result in the impossibility of passing the course, and the student will then be subject to the appropriate disciplinary procedures indicated in the ACTION PROTOCOL FOR FRAUDULENT PRACTICES AT THE UNIVERSITY OF VALENCIA (ACGUV 123/2020).

REFERENCES

- Lenguajes de programacion. Principios y paradigmas, Allen B. Tucker, Robert Noonan, McGraw-Hill/Interamericana de España, S.A.U., 2003
- Java a tope: Compiladores, Sergio Gálvez Rojas y Miguel Ángel Mora Mata, Universidad de Málaga. Disponible en Web: http://www.giaa.inf.uc3m.es/docencia/II/PL2/herramientas/JFlex_JavaCC.pdf
- Functional Programming for Java Developers, Dean Wampler, OReilly Media, Inc, 2011. ISBN 978-1-4493-1103-2
- On LISP, Advanced Techniques for Common LISP, Paul Graham, Prentice Hall, 1993, ISBN 0130305529. Versión gratuita en http://paulgraham.com/onlisp.html
- The Python Language Reference. Release 3.12.4. Guido van Rossum and the Python development team. Versión gratuita en https://fossies.org/linux/python-docs-pdf-a4/reference.pdf
- Functional Programming in Python. David Mertz, O'Reilly Media, Inc. ISBN 978-14919-2855-4
- PLY Reference Manual. David M. Beazley. Versión Gratuita https://www.dabeaz.com/ply/ply.html
- LLVM Essentials. Mayur Pandey, Suyog Sarda. O'Reilly. ISBN: 978-17852-8080-1