Universidad de Alcalá

TEACHING GUIDE

Fundamentals of Programming

Degree in
Computer Engineering (GIC)
Computer Science Engineering (GII)
Information System Engineering (GISI)

Universidad de Alcalá

Academic Year 2022/2023

1st Year - 1st Semester (GIC+GII+GISI)

Approved by the School Board on June 3, 2022

TEACHING GUIDE

Course Name:	**Fundamentals of Programming**
Code:	**780003 (GIC+GII+GISI)**
Degree in:	**Computer Engineering (GIC)** **Computer Science Engineering (GII)** **Information System Engineering (GISI)**
Department and area:	**Ciencias de la Computación** **Computer Science**
Type:	**Basic (GIC+GII+GISI)**
ECTS Credits:	**6.0**
Year and semester:	**1st Year - 1st Semester (GIC+GII+GISI)**
Teachers:	Salvador Sanchez Alonso
Tutoring schedule:	Will be made public on the first session
Language:	English

1. COURSE SUMMARY

This subject aims to introduce students in the world of programming, providing the core knowledge to begin programming in any language, with no assumed previous knowledge. The course is divided in 2 parts, the laboratory part provides basic skills in Python programming, while the theoretical part focuses on software design with an emphasis on understanding the theory underlying the use of programming languages.

The subject teaches and illustrates the software design process, and shows how to develop a correct, readable and reusable solution from a problem specification.

2. SKILLS

Basic, Generic and Cross Curricular Skills.

This course contributes to acquire the following basic, generic and Cross Curricular skills:

en_CB1 - That students have demonstrated to possess and understand knowledge in an area of study that is based on general secondary education, and is usually found at a level that, although supported by advanced textbooks, also includes some aspects that involve knowledge from the forefront of their field of study.

en_CB2 - That the students know how to apply their knowledge to their work or vocation in a professional manner and possess the competencies that are usually demonstrated through the elaboration and defense of arguments and the resolution of problems within their area of study.

en_CB3 - That students have the ability to gather and interpret relevant data (usually within their area of study) to make judgments that include a reflection on relevant social, scientific or ethical issues.

en_CB4 - That students can transmit information, ideas, problems and solutions to both a specialized and non-specialized public.

en_CB5 - That the students have developed those learning skills necessary to undertake further studies with a high degree of autonomy.

en_CG5 - Ability to conceive, develop and maintain computer systems, services and applications using software engineering methods as an instrument for quality assurance, in accordance with the knowledge acquired as established in section 5, annex 2, of the resolution BOE-A-2009-12977.

en_CG8 - Knowledge of the basic subjects and technologies, which enable them to learn and develop new methods and technologies, as well as those that provide them with great versatility to adapt to new situations.

en_CG9 - Ability to solve problems with initiative, decision making, autonomy and creativity. Ability to know how to communicate and transmit the knowledge, skills and abilities of the profession of Computer Engineering Engineer.

en_TRU1 - Capacity of analysis and synthesis.

en_TRU2 - Oral and written competencies.

en_TRU3 - Ability to manage information.

en_TRU4 - Autonomous learning skills.

en_TRU5 - Team work.

Specific Skills

This course contributes to acquire the following professional skills:

en_CIB3 - Ability to understand and master the basic concepts of discrete mathematics, logic, algorithmic and computational complexity, and its application for solving engineering problems.

en_CIB4 - Basic knowledge of the use and programming of computers, operating systems, databases and computer programs with application in engineering.

en_CIB5 - Knowledge of the structure, organization, operation and interconnection of computer systems, the fundamentals of their programming, and their application for solving engineering problems.

Learning Outcomes

After succeeding in this subject the students will be able to

RA1. Develop the ability to create algorithmic solutions to problems and be able to represent them in the form of computer programs.
RA2. Apply the top-down implementation strategy and the principles of modular design to the construction of programs, following the principles of maximum cohesion and minimum coupling.
RA3. Distinguish between different development, design, test and debugging techniques applied to problems, acquiring a systemic vision of verification and validation.
RA4. Experiment with a high-level programming language and environment, identifying its capabilities and limitations compared to other languages â€‹â€‹and environments.
RA5. Explain the basic concepts of data storage and representation, and deduce both the structure of a data and its compatibility with others based on its type.
RA6. Recognize the verification tasks that are carried out during the processing of a program, and design programs with them in mind.

3. CONTENTS

Contents Blocks	Total number of hours *
Programming fundamentals. Introduction, syntax and semantics of high-level programming languages. Basic concepts: variables, types, expressions and assignments. Basic input / output. Selective and iterative control structures. Modularization: functions and parameter use, modular decomposition of programs.	16 hours
Software and Algorithms development methodology and problem solving. Problem solving strategies: the role of algorithms in the problem solving process, implementation of strategies for algorithms, testing and debugging of the code, concept of algorithm and properties of the same.	14 hours
Basic data structures. Internal representation of the data; primitive types; structured types; definition and use of new types.	16 hours
Recursion. Concept, classification of recursion, transformation of recursive algorithms, classic examples.	8 hours

* 58 hours in total: 56h classes + 2 hours assessment

4. TEACHING - LEARNING METHODOLOGIES. FORMATIVE ACTIVITIES.

4.1. Credits Distribution

Number of on-site hours:	58 hours (56 hours on-site +2 exams hours)
Number of hours of student work:	92
Total hours	150

4.2. Methodological strategies, teaching materials and resources

The course is organized as a 6 ECTS four-month course, using the following training activities in the teaching-learning process of its contents:

- Theoretical classes in person.
- Face-to-face practical classes.
- Face-to-face laboratory practices.
- Mentoring: individual and / or group.

In addition, depending on the nature of the different parts of the subject under study, the following training activities may be used, among others: Preparation of individual or team work, sharing of information, problems and doubts that appear in the performance of the work and use of the Virtual Classroom Platform.

5. ASSESSMENT: procedures, evaluation and grading criteria

Preferably, students will be offered a continuous assessment model that has characteristics of formative assessment in a way that serves as feedback in the teaching-learning process.

5.1. PROCEDURES

The evaluation must be inspired by the criteria of continuous evaluation (Learning Assesment Guidelines, LAG, art 3). However, in compliance with the regulations of the University of Alcalá, an alternative process of final evaluation is made available to the student in accordance with the Learning Assesment Guidelines as indicated in Article 10, students will have a period of fifteen days from the start of the course to request in writing to the Director of the Polytechnic School their intention to take the non-continuous evaluation model adducing the reasons that they deem convenient. The evaluation of the learning process of all students who do not apply for it or are denied it will be done, by default, according to the continuous assessment model. The student has two calls to pass the subject, one ordinary and one extraordinary.

Ordinary Call

Continous Assessment:

In the ordinary call, the default evaluation method is continuous evaluation, with formative evaluation characteristics to serve as feedback in the teaching-learning process by the student

Assessment through final exam:

Optionally, and with justification, the student may request the evaluation by a single test before the director of the center, which must be requested in writing and within the regulated deadlines. This exam will consist of a single, written test, where all the contents of the subject will be evaluated, both the theoretical part and the laboratory part.

Extraordinary Call

The procedure will be the same as that described for the assessment by means of a final exam in the ordinary call.

5.2. EVALUATION

EVALUATION CRITERIA

The following criteria will be used for the evaluation of the subject, taking into account the degree of acquisition of the competences by the student:

CE1. The student demonstrates the mastery of the basic rudiments of programming by writing simple, complete, robust and efficient code.

CE2. The student demonstrates aptitude in solving problems using computer programs.

CE3. The student distinguishes the different control structures of the logic of a program and is able to use the most appropriate in each circumstance.

CE4. From the description of a complex problem, the student is able to design modular programs whose structure is determined by the principles of cohesion and coupling according to the good practices of top-down programming.

CE5. The student is capable of designing iterative and recursive solutions for the same problem and distinguishes the advantages and disadvantages of each one.

CE6. The student is able to carry out a correct analysis of the necessary data types and structures, as well as to propose design alternatives based on the specific needs of the problem.

CE7. The student understands a code written by another person and is able to modify it to correct it or to extend its functionality.

ASSESSMENT INSTRUMENTS

This section summarizes the evaluation instruments that will be applied to each of the evaluation criteria.

- Intermediate Assessment Test (PEI-1): Resolution of theoretical-practical questions on control structures, rudiments of modular programming and basic concepts.
- Laboratory Test (PL-1): Preparation of simple programs in a programming language.
- Intermediate Assessment Test (PEI-2): Resolution of theoretical-practical questions on structured data, recursion, handling of data structures and data processing algorithms.
- Laboratory Test (PL-2): Elaboration of programs of intermediate complexity in a programming language.
- Final Assessment test (PEF): Consisting of solving theoretical-practical assumptions, as well as solving complex problems through the use of programs in a programming language.

ASSESSMENT CRITERIA

a) Ordinary Call

Competence	Learning outcome	Assessment criteria	Assessment instrument	Weight
CG8, CG9, CIB4, CB1-2, CB5, TRU1-4	RA1, RA2, RA5	C1, C3, C7	PEI-1	10%
CG8, CG9, CIB4, CB1-2, CB5, TRU1-5	RA1, RA2, RA4, RA6	C1, C2, C3, C7	PL-1	10%
CG5, CG8, CG9, CIB3-5, CB3-4, TRU1-4	RA1, RA2, RA3, RA4, RA5, RA6	C1, C2, C3, C5, C7	PEI-2	40%
CG5, CG8, CG9, CIB3-5, CB3-4, TRU1-5	RA1, RA2, RA3, RA4, RA6	C1, C2, C3, C4, C5, C6	PL-2	40%

b) Extraordinary Call / Ordinary call - Final assessment

Competence	Learning outcome	Assessment criteria	Assessment instrument	Weight
CIB3-5, CG5, CG8, CG9, CB1-5, TRU1-5	RA1, RA2, RA3, RA4, RA5, RA6	CE1, CE2, CE3, CE4, CE5, CE6, C7	PEF	100%

All the assessment tests can be carried out in the theory or laboratory classrooms, or through the Virtual Classroom. A pass mark in the practices and the activities of the subject is a necessary requirement for passing the subject.

As a result of the assessment process, the students will obtain a grade that will depend on their activity in the different tests of the subject. The result of each test will yield information either through quantitative indicators of acquisition of skills.

6. BIBLIOGRAPHY

6.1. Basic Bibliography

- MARZAL, A. y GRACIA, I. Introducción a la programación con Python - UJI. Ed. Publicacions de la Universitat Jaume I.
- PEÑA, R. Resolución de problemas para ingenieros con Python estructurado. Ed. Garceta.
- SEVERANCE, C. Python for informatics: http://www.pythonlearn.com/book.php

6.2. Additional Bibliography

a) Python

- Learn Python the hard way - http://learnpythonthehardway.org/book/
- A bite of Python - http://www.swaroopch.com/notes/python/
- Think Python - http://www.greenteapress.com/thinkpython/
- John M. Zelle (2010) Python Programming: An Introduction to Computer Science. Editorial Franklin, Beedle & Associates. 2ª ed.

b) Theory of programming:

- GARCÍA MOLINA, F., MONTOYA DATO, J. y otros. Una Introducción a la Programación: Un enfoque algorítmico. Ed. Thomson Paraninfo. 2005.
- JOYANES, L. Fundamentos de la programación, 1ª Ed. Ed. McGraw-Hill. 1992.

Disclosure Note

The University of Alcalá guarantees to its students that, if due to health requirements the competent authorities do not allow the total or partial attendance of the teaching activities, the teaching plans will achieve their objectives through a teaching-learning and evaluation methodology in online format, which will return to the face-to-face mode as soon as these impediments cease.