Programming

Academic Year: (2022 / 2023) **Review date: 13-05-2022**

Department assigned to the subject: Computer Science and Engineering Department

Coordinating teacher: GARCIA OLAYA, ANGEL

Type: Basic Core ECTS Credits : 6.0

Year : 1 Semester : 1

Branch of knowledge: Engineering and Architecture

REQUIREMENTS (SUBJECTS THAT ARE ASSUMED TO BE KNOWN)

None

OBJECTIVES

Specific objectives:
- Knowledge of fundamentals of imperative programming from an object oriented point of view.
- Basic knowledge of the syntax of an object oriented programming language.
- Knowledge of best programming practices and code style
- Ability to break down a real problem by following an object oriented methodology, in order to code it into a computer program.
- Knowledge of simple structures for information management.
- Ability to understand technical documents and to reuse third parties programming code and libraries.
- Basic knowledge of computational complexity and search and sort algorithms.

General objectives:
- Ability to self-organize and individual work and learning process planning.
- Ability to work in groups.
- Oral and writing skills.
- Analysis and synthesis abilities.
- Ability to decide among alternatives.
- Motivation for quality.

DESCRIPTION OF CONTENTS: PROGRAMME

1. Introduction
   1.1. Components of a program: data and algorithms
   1.2. Computer architecture
   1.3. Data storage and coding
   1.4. Brief introduction to the history of programming
   1.5. Compilation vs. Interpretation
   1.6. Programming paradigms: imperative, logical and functional programming
2. Flow diagrams
3. Introduction to Python
   3.1. The Python interpreter
   3.2. Basic data types
   3.3. Programs
   3.4. Variables and constants
   3.5. Input, output and random numbers
   3.6. Operators
   3.7. Casting
   3.8. Comments
   3.9. Debugging: errors
4. Flow control: conditionals and loops
   4.1. Conditionals
   4.2. Scope and blocks
   4.3. Loops
5. Simple data structures
   5.1. Lists and tuples
   5.2. Dictionaries
6. Functions

6.1. Decomposition
6.2. Code reuse
6.3. Implementation hiding
6.4. Pass by value and pass by reference
6.5. Introduction to recursion
7. Introduction to Object Oriented Programming
7.1. Classes and objects
7.2. Methods
7.3. Inheritance
7.4. Polimorphism
8. Algorithms
8.1. Introduction to computational complexity
8.2. Search
8.3. Sorting

LEARNING ACTIVITIES AND METHODOLOGY

1. Lectures (1 ECTS) Devoted to teaching basic competences, especially those related to basic imperative programming techniques
2. Lab sessions (1 ECTS) They complement the lectures and allow the students to put their knowledge into practice
3. Autonomous work (2.5 ECTS) Devoted to individual or group learning the specific competences. This activity will also contribute to the following generic competences: Oral and writing skills, Analysis and synthesis abilities, Ability to decide among alternatives, Motivation for quality and Ability to self-organize, and work and learning process planning capacity.
4. Office hours (1 ECTS). Individual or in group office hours as requested by the students
5. Final exam (0.5 ECTS) Development with limited time of an object oriented program. It allows the student to show his/her overall knowledge

METHODOLOGY

Classes will be divided into lectures and laboratory sessions. In addition, students are expected to work autonomously at home.

- Lectures: they will be used to present general programming concepts using Python as language. Example programs will be written during the class, to reinforce the theoretical concepts. Short test can be performed at the end of each class to evaluate the understanding of the main concepts presented in that class and the previous ones.

- Laboratory classes: they will consist of three parts. In the first part, the students, randomly selected, will be asked to explain the previous week proposed exercises. Next, a problem will be posed and solved in collaboration by students and teacher. Finally, a series of problems will be presented for the students to solve them. These problems need to be uploaded to Aula Global by the end of the week. A final project will be presented to be performed in groups of two. It could include partial deliveries. Both the final project and the weekly assignments will be performed in Python.

- Student's autonomous work: students are expected to finish at home the weekly assignments and to deliver them using Aula Global.

Office hours
Following university rules each professor will publish his/her preferred office hours in Aula Global. Despite this, students are encouraged to contact the professor by email to arrange an appointment any time they want, whether in the preferred hours or not.
Small doubts not requiring a face-to-face visit can be made by email. General doubts should be posted in the course forum, available in Aula Global. These doubts can be solved either by the professors or by other students.
For the final project, individual tutorship sessions will be arranged for each group.

ASSESSMENT SYSTEM

Combined and continuous evaluation of every activity performed by the student either individual or in group, considering the effort devoted by each student to each of the formative activities described.
A formative evaluation through continuous feedback will be performed, allowing the student to know which is his/her knowledge and what it is expected from him/her. Evaluation will be composed of:
- Mid-term exams and tests at the end of classes: 1.5 points
- Individual oral presentations of the exercises: 0.5 points
- Weekly assignments: 0.5 points (if at least 80% of them are presented)
- Final project: 2.5 points

- Final exam, marked with 5 points

To pass the subject, a total of 5 points are needed, which will be calculated in the following way, depending on the call:

In the January call the final mark will be the maximum between:
1) 60% of the mark of the final examination (out of 10 points maximum)
and
2) continuous evaluation mark + final examination mark / 2

To apply the formula of item 2), the mark of the final examination must be greater or equal than 4 out of 10. In case of plagiarism in any of the practical exercises, all the involved students will loose the right to follow continuous evaluation and formula 1) will be applied.

In the extraordinary call the final mark will be the maximum between:

1) the mark of the final examination (out of 10 points maximum)
and
2) continuous evaluation mark + final examination mark / 2

As in the January period, to apply the formula of item 2), the mark of the final examination must be bigger or equal than 4 out of 10

It will not be possible to present any kind of practical exercises in the extraordinary exams period.

Individual activities will account for a 75% of the final mark, group activities will constitute the remaining 25%. Continuous evaluation will represent a 50% of final mark, while final examination will account for the remaining 50%.

% end-of-term-examination:	50
% of continuous assessment (assigments, laboratory, practicals…):	50

BASIC BIBLIOGRAPHY

- Ana Bell Get Programming Learn to code with Python, Manning publications, 2018
- John S. Conery Explorations in Computing: An Introduction to Computer Science and Python Programming, CRC Press, 2014

BASIC ELECTRONIC RESOURCES

- Guido van Rossum, Barry Warsaw, Nick Coghlan . PEP 8 -- Style Guide for Python Code: https://www.python.org/dev/peps/pep-0008/
- Python Software Foundation . Python for Beginners: https://www.python.org/about/gettingstarted/