Course guide
240715 - 240715 - Programming

Last modified: 29/08/2023

Unit in charge: Barcelona School of Industrial Engineering
Teaching unit: 723 - CS - Department of Computer Science.

Degree: BACHELOR'S DEGREE IN INDUSTRIAL TECHNOLOGIES AND ECONOMIC ANALYSIS (Syllabus 2018). (Compulsory subject).

Academic year: 2023 **ECTS Credits:** 6.0 **Languages:** English

LECTURER

Coordinating lecturer: Vila Grabulosa, Lluís

Others: Grup: 11: Vila Grabulosa, Lluís
Grup: 12: Vila Grabulosa, Lluís

PRIOR SKILLS

English language, both written and conversational.
User-level computer skills (including good keyboard typing).

REQUIREMENTS

Installing the Anaconda platform http://www.anaconda.com/ at your personal computer. go to https://docs.anaconda.com/free/anaconda/install/index.html/

TEACHING METHODOLOGY

Classes combine the following pedagogical, practice-oriented activities all based on programming Python functions using a common Python IDE (Integrated Development Environment):

1.- Very short lectures: Practical lectures on Python programming ideas, language components and methodological recipes all illustrated by hands-on programming using the Python IDE (everything shown to the class using the classroom beamer) interleaved with examples for the students to try and experiment with using the same IDE.

2.- Practice: Student practice on solving small problems using the Python IDE under the professor supervision and guidance.

3.- Student Presentations: Optional, upon request, short (15' max) lectures exposed by student teams (2 students max) on popular Python modules such as Pandas or Numpy (a full list of options will be published in the Atenea course). Students are encouraged to attend the lectures and pose questions (the questions round is 5' max).

4.- Tests: Short individual tests on topics recently studied in class, including the student lectures previously mentioned.

Classes are organized as follows:

- Two classes per week in a computer classroom.
- All classes are two hours long.
- All classes start at 5 minutes past the hour o'clock, finish 5 minutes before the hour and do not include breaks yet students are allowed to leave the classroom anytime they need.
- The time devoted to each type of activity is not fixed a priori but is variable depending on the development of the class.
- The number of tests is not less than 4 and no more than 12.

LEARNING OBJECTIVES OF THE SUBJECT

Main skills:
- Learning the basic concepts and processes of computer programming.
- Programming small Python functions to solve small problems.
- Develop basic skills in the main phases of computer programming skills, namely: problem specification, simple program analysis, coding, testing via doctests and debugging.
- Using an Integrated Development Environment.

Additional skills:
- Planning and following a study calendar.
- Preparing and performing a public presentation of a technical subject: including: oral expression, slides design, technical report writing.
- Team working.

STUDY LOAD

Type	Hours	Percentage
Hours medium group	60,0	40.00
Self study	90,0	60.00

Total learning time: 150 h

CONTENTS

Introduction

Description:
Algorithm, program, programming language
Compiled and interpreted languages
Python interpreter (shell)
Shell scripts
Values and types
Basic types: integer numbers (int), float numbers (float), characteres, strings (str), booleans (bool)
Expressions, operands, operators
Comparison operations
Variables: name, type and value
Assignment statement
Sequential composition
Input/Output
Program comments

Full-or-part-time: 10h
Practical classes: 4h
Self study : 6h

Functions

Description:
Syntax: def, parameters, body, indentation.
Function: definition and call.
Formal and actual parameters, local variables.
Name space, scope.

Full-or-part-time: 10h
Practical classes: 4h
Self study : 6h

Conditionals

Description:
Boolean expression (condition)
Simple Conditional statement.
(full) Conditional Statement.
Sequencial and nested conditionals.

Full-or-part-time: 10h
Practical classes: 4h
Self study : 6h

Strings

Description:
Definition: characters, ASCII table, type str, empty string, valid string.
Functions: len( )
Boolean operators: in, not in, ==, Construction operators: +, *
Indexing. Access operator [ ]
Slice operator [:]
Methods: .find, .count, .index, .replace, .reverse, ...
Properties: Immutability.
Route: for ... in ... :

Full-or-part-time: 10h
Practical classes: 4h
Self study : 6h

Lists

Description:
Definition and commonalities with strings:
- Funcions: len( )
- Operacions: +, *, in, not in, ...
- Acces and Slice:
- Methods: .count, .index,
- Traversal
String-List mehods: .split, .join, .sorted,
Heterogeneity property.
Imbrication property.
Mutability property:
- Modifier methods: .append .extend. sort .reverse
- Deleting elements
- Aliasing and cloning
- Function parameters and aliasing
List index traversal
- Index generation functions: range( ), enumerate( ).
- Application

Full-or-part-time: 28h 45m
Practical classes: 11h 30m
Self study : 17h 15m

Tuples

Description:
Definition and commonalities with lists
Immutability property
Applicability

Full-or-part-time: 1h 15m
Practical classes: 0h 30m
Self study : 0h 45m

Dictionaries

Description:
Definition: key:value pairs
Indexing (key)
Operations: [ ], len, in, not in
Methods: .items, .keys, .values, .get
Dictionary traversal
Mutability, aliasing and cloning

Full-or-part-time: 10h
Practical classes: 4h
Self study : 6h

Files

Description:
Sequential text files
The "representative" of a file
File opening, modes:
- open( ) function
- with ... as , ... : instruction
- Methods: .read, .readline, .readlines, .write
- Lines traversal

Full-or-part-time: 10h
Practical classes: 4h
Self study : 6h

Iterations with while

Description:
The while statement:
- syntax: comparison with if and for.
- semantics
while versus for
Infinite loops
Iteration counter

Full-or-part-time: 20h
Practical classes: 8h
Self study : 12h

Libraries

Description:
Student Presentation made by teams (2 or 3 people) of a Python library to choose upon the following:
- Pandas
- NetworkX
- Matplotlib
- Seaborn
- Tkinter
- Scrapy
- Flask
- Kivy
- NumPy
- SciPy
- Sci-Kit Learn
- TensorFlow
- PyTorch
- Keras
- OpenCV
- Pygame

Full-or-part-time: 10h
Practical classes: 4h
Self study : 6h

Problem Solving

Description:
Problem Solving

Full-or-part-time: 10h
Practical classes: 4h
Self study : 6h

GRADING SYSTEM

The course grade is based on the following evaluation activities:

1.- Tests:
- Eval: Average of all the grades but one (the lowest one) from the tests done in class. Tests not attended by the student will be computed as 0.0 unless a written justification is presented.
- Grade: Ranges from 0.0 to 10.0.
- Schedule: Tests are scheduled throughout the lective weeks of the course during class hours.
- Organization: Previous registration is not necessary.

2.- Presentations:
- Eval: Average of the oral presentation, quality of the slides, and the enclosed short report.
- Grade: Ranges from 0.0 to 2.0. This grade is *optional* (meaning that the course grade can still reach the 10.0 maximum without doing any).
- Schedule: Student lectures will be scheduled during the last weeks of the lective period and will be done during class hours.
- Organization: All students can prepare a lecture upon request only by both members of the students team.

3.- Final Exam:
- Eval: Long exam (between 3 and 5 questions in 2 hours max) covering all the contents of the course including the student lectures.
- Grade: Ranges from 0.0 to 10.0.
- Schedule: this exam is scheduled during the last week of the lective period outside class hours.
- Organization: All students are allowed to attend this exam. Previous registration is no necessary.

4.- Re-evaluation Exam:
- Eval: Long exam (between 3 and 5 questions in 2 hours max) covering all the contents of the course.
- Grade: Ranges from 0.0 to 10.0.
- Schedule: This exam scheduled during the January exam period (possibly under the "final exam" label).
- Organization: All students are allowed to attend this exam, but it is mandatory to register beforehand.

The Course Grade is computed according to the following rules:

- Rule #1: Course Grade = 0.5 * Tests Grade + 0.5 * Final Grade

- Rule #2: If the student passed the Course Grade (grade >= 5.0) and he/she hass done a presentation then
Course Grade += presentation grade (with a max of 10.0).

- Rule #3: If the student has done the Re-evaluation exam then
Course Grade = Re-evaluation Grade. In this case Rule #2 equally applies as well.

EXAMINATION RULES.

As specified in the Atenea course.

RESOURCES

Other resources:
As specified in the Atenea course.