SUBJECT TEACHING GUIDE

G40 - Programming

Degree in Physics

Academic year 2018-2019

1. IDENTIFYING DATA					
Degree	Degree in Physics			Type and Year	Compulsory. Year 1
Faculty	Faculty of Sciences				
Discipline	Subject Area: Computation Tools Central Module				
Course unit title and code	G40 - Programming				
Number of ECTS credits allocated	6	Term		Semester based (2)	
Web	https://www.istr.unican.es/asignaturas/prog_python				
Language of instruction	Spanish	English Friendly	Yes	Mode of delivery	Face-to-face

Department	DPTO. INGENIERÍA INFORMÁTICA Y ELECTRÓNICA
Name of lecturer	MICHAEL GONZALEZ HARBOUR
E-mail	michael.gonzalez@unican.es
Office	Facultad de Ciencias. Planta: + 3. DESPACHO PROFESORES (3055)
Other lecturers	JOSE JAVIER GUTIERREZ GARCIA JOSE IGNACIO ESPESO MARTINEZ JOSE CARLOS PALENCIA GUTIERREZ ADOLFO GARANDAL MARTIN

3.1 LEARNING OUTCOMES

- • To be able to design, implement and test algorithms and simple programs using an object-oriented language.
  • To know how to code classes and objects by applying object-oriented programming.
  • To know basic algorithms applicable to structured data (such as iteration, searches, matrix operations)
  • To use a programming environment to edit, compile and run programs.
  • To know how to use operating systems to perform basic tasks from the command line.

4. OBJECTIVES

- To know and understand the syntax and semantics of the statements of an imperative programming language.
- To know the main algorithmic constructs: sequence, conditionals, iteration and recursion
- To understand the concepts of class and object as constituent elements of the programs
- To know and use basic data types, tables and matrices, and to learn basic algorithms for their manipulation (iteration, searches, simple sorting).
- To understand the concepts of method and parameter passing.
- To know the principles of modularity and abstraction to create simple program modules
- To know how to perform error handling with exceptions
- To acquire basic knowledge of object-oriented programming
- To know the principles of input / output: interactive and with files
- To use basic operating system commands

- To design small algorithms using pseudocode.
- To be able to code and test algorithms using an imperative programming language.
- To use a development system to edit, compile and run programs.
- To use an operating system at the user level.
- To create program modules, separating the phases of design and implementation.
- To code programs using an object-oriented language.
- To implement simple programs that are reliable and easy to understand.
- To use predefined program modules.
- To apply simple test strategies for a program module.
- To know how to document a programming project.

6. COURSE ORGANIZATION

CONTENTS	
1	THEMATIC UNIT 1: Programming in Python
1.1	0. Introduction to the course
1.2	1. Introduction to programming languages. High-level languages. Compilers and interpreters. The software life cycle. Concept of algorithm. Program structure. Functions. Coding style.
1.3	2. Data and expressions. Numbers and basic operations. Variables. Booleans. Strings. Lists. Operators and expressions. Using mathematical functions.
1.4	3. Classes. Classes and objects. Class diagrams. Instance attributes and methods. Static and class attributes and methods. Name spaces. Modules and packages.
1.5	4. Algorithmic structures. Conditional instruction. Multiway branch statement. Loop statements. Iterators. Recursion. Describing algorithms with pseudocode.
1.6	5. Data structures. Lists and unidimensional tables. Iteration and search algorithms. Tuples, sequences, sets and dictionaries. The Numpy package. The Scipy package.
1.7	6. Error handling. Exceptions. Exception handling. Exception handling patterns. Throwing exceptions. Using our own exceptions. Cleaning actions.
1.8	7. Input / Output. Input / output of text and numbers. The Matplotlib package. Formatted text output. Files. Writing text files. Reading text files.
1.9	8. Inheritance and polymorphism. Inheritance. Abstract classes. Programming by extension and incremental programming. Polymorphism.
2	THEMATIC UNIT 2: Tools
2.1	9. Using operating systems. Basic concepts. Common operating systems.The shell. Running programs. The graphical file manager. Using the USB memory. Scripts.
2.2	10. Using an integrated software development environment. Program development process. Installation. Using the Python interpreter. Debugging. Document generation.

7. ASSESSMENT METHODS AND CRITERIA

Description	Type	Final Eval.	Reassessn	%
Presentation of solutions to short exercises solved at home	Others	No	Yes	10,00
Notes and books are allowed during the final exam, but not electronic media such as laptops, tablets and mobile phones.	Written exam	Yes	Yes	50,00
Theory: classroom participation	Others	No	No	10,00
Assessment of lab assignments and reports. There will be penalties for late reports.	Laboratory evaluation	No	Yes	30,00
TOTAL				100,00

Observations

The grade for the course consists of three parts:

a) classroom participation: 10%
b) continuous assessment of exercises and lab assignments: 40%
This part of the assessment consists of two parts:
b.1) Short exercises (10%)
b.2) Lab assignments (30%)
c) Final exam: 50%

Part b, continuous assessment of exercises and lab assignments, can be replaced by a lab exam in September.

To pass the subject is necessary to score a minimum mark of 4 in both part b) and part c). Should one of these parts not be passed with the minimum mark, the final mark will be the minimum of 4.5 and the average obtained. If only one of these parts is passed in the ordinary evaluation period, the mark obtained for that part will be saved for the extraordinary evaluation period.

Observations for part-time students

For part-time students continuous assessment of short exercises and lab assignments can be replaced by a lab exam, both in the ordinary and extraordinary evaluation periods: For these students the weight of this exam is 40%, and the weight of the final exam is 60%. There will be no evaluation of the classroom participation.

8. BIBLIOGRAPHY AND TEACHING MATERIALS

BASIC

Title: Python in a Nutshell: A Desktop Quick Reference 3rd Edition
by: Alex Martelli (Author), Anna Ravenscroft (Author), Steve Holden
Publisher: O'Reilly Media; 3 edition (May 4, 2017)
ISBN-10: 144939292X
ISBN-13: 978-1449392925

Tutorial de python 3:
https://docs.python.org/3/tutorial/
http://docs.python.org.ar/tutorial/3/index.html

Title: Introducción a la programación con Python 3
By: Andrés Marzal Varó, Isabel Gracia Luengo, Pedro García Sevilla
Editor: Universitat Jaume I, 2014
ISBN: 978-84-697-1178-1
http://repositori.uji.es/xmlui/bitstream/handle/10234/102653/s93.pdf