Course guides
330053 - I - Introduction to Computing

Last modified: 05/05/2020

Unit in charge: Manresa School of Engineering
Teaching unit: 750 - EMIT - Department of Mining, Industrial and ICT Engineering.

Degree: BACHELOR'S DEGREE IN ELECTRICAL ENGINEERING (Syllabus 2009). (Compulsory subject).
BACHELOR'S DEGREE IN INDUSTRIAL ELECTRONICS AND AUTOMATIC CONTROL ENGINEERING (Syllabus 2009). (Compulsory subject).
BACHELOR'S DEGREE IN MECHANICAL ENGINEERING (Syllabus 2009). (Compulsory subject).
BACHELOR'S DEGREE IN CHEMICAL ENGINEERING (Syllabus 2009). (Compulsory subject).
BACHELOR'S DEGREE IN ENERGY AND MINING RESOURCE ENGINEERING (Syllabus 2012). (Compulsory subject).
BACHELOR'S DEGREE IN INDUSTRIAL ELECTRONICS AND AUTOMATIC CONTROL ENGINEERING (Syllabus 2016). (Compulsory subject).
BACHELOR'S DEGREE IN MECHANICAL ENGINEERING (Syllabus 2016). (Compulsory subject).
BACHELOR'S DEGREE IN CHEMICAL ENGINEERING (Syllabus 2016). (Compulsory subject).

Academic year: 2020 **ECTS Credits:** 6.0 **Languages:** Catalan

LECTURER

Coordinating lecturer: Vives Pons, Jordi

Others: Llusà Serra, Aleix
Arumi Casanovas, Arnau
Masip Riera, Jordi

DEGREE COMPETENCES TO WHICH THE SUBJECT CONTRIBUTES

Specific:
1. Basic knowledge on the use and programming of computers.
2. Operating systems, databases and computer programs with applications in engineering.

Transversal:
3. EFFICIENT ORAL AND WRITTEN COMMUNICATION - Level 1. Planning oral communication, answering questions properly and writing straightforward texts that are spelt correctly and are grammatically coherent.
4. TEAMWORK - Level 1. Working in a team and making positive contributions once the aims and group and individual responsibilities have been defined. Reaching joint decisions on the strategy to be followed.
5. EFFECTIVE USE OF INFORMATI0N RESOURCES - Level 1. Identifying information needs. Using collections, premises and services that are available for designing and executing simple searches that are suited to the topic.
6. SELF-DIRECTED LEARNING - Level 1. Completing set tasks within established deadlines. Working with recommended information sources according to the guidelines set by lecturers.

TEACHING METHODOLOGY

The subject is divided into two 2-hour classes per week. Of the four face-to-face hours, the first is devoted to introducing the main topics in lectures, the second to solving the problems posed by students and the remaining two to solving practical problems in the computer laboratory. Every week, students are told what they need to study and the problems they must solve. It is advisable for students to do these exercises in groups, at least partially. Students' individual progress is periodically assessed. The subject also includes a medium-sized software development project that must be carried out in groups.

LEARNING OBJECTIVES OF THE SUBJECT

On completion of the subject, students must be able to:
1. Apply the fundamental concepts of computer programming.
2. Show mastery of basic programming techniques and tools.
3. Solve problems by developing programs of low- and mid-level complexity.
4. Demonstrate abstract thinking in using models to solve real problems.
5. Plan oral presentations, correctly reply to questions and write basic texts without spelling and grammar mistakes.
6. Recognise one's own information needs and use the collections, spaces and services available to design and execute simple searches related to a specific field.
7. Complete tasks in the time allotted, using the suggested materials and following the guidelines set by the professor.

STUDY LOAD

Type	Hours	Percentage
Hours small group	30,0	20.00
Hours large group	30,0	20.00
Self study	90,0	60.00

Total learning time: 150 h

CONTENTS

Topic 1. Introduction to programming

Description:
Students must dive in to the world of programming without hesitation. Therefore, they are introduced to its most basic concepts straight away, without going into detail, so that they can quickly get acquainted with it. A solid understanding is not sought at first; rather, in experimenting, students acquire a series of tools that will later enable them to progress more rapidly.
Keywords: Computer, program, algorithm, programming error (bug), programming language, portability, interpreter, shell, script, debugging, syntax errors, execution errors, semantic errors, value, variable, type, allocation, input, output, reading, writing, statement, reserved word, expression, operator, operand, precedence, evaluation, composition of statements, function, header, body, call, parameters, return value, locality.

Related activities:
As stated.

Full-or-part-time: 12h
Laboratory classes: 4h
Self study : 8h

Topic 2. Conditionals and iterations

Description:
Competencies of the degree to which the subject contributes
Description: This topic continues from the previous one in the same vein: a quick, practical and intuitive stroll through the basics. The aim is not for students to gain in-depth knowledge but rather a superficial take on the topic. The basics studied are essentially alternative and iterative constructs.
Keywords: Boolean modules, types and values, Boolean expressions, Boolean operators, alternative (or conditional) statements, block, chained and nested statements, function return, reading keyboard input, type casting, None, composition of functions as expressions, Boolean functions, functions as values, iterative statement, infinite iterations, local variables.

Related activities:
As stated

Full-or-part-time: 12h
Laboratory classes: 4h
Self study : 8h

Topic 3. Documentation and program testing

Description:
This topic eases the pressure after the previous two, as it simply introduces the concept of the unit test and doctest and nose tools. The aim is two-fold: first, the knowledge acquired in previous topics must be consolidated, and second, the knowledge of unit testing via doctest that will allow students to use this tool without problems from this topic onwards must be acquired.
Keywords: Unit test, doctest, nose, test-driven development.

Related activities:
As stated

Full-or-part-time: 6h
Laboratory classes: 2h
Self study : 4h

Topic 4. Strings

Description:
The first three topics ran through the basics of a programming language and focused on their immediate application. They make up the first subject area of the course. This new subject area adds the basics of information storage and builds on the previous topics. Once this second subject area has been completed, the power of the elements available to students is considerable. This topic is devoted to strings. In addition to their inherent interest, they represent two important topics in Python: sequences and immutable types.
Keywords: String, structural type (as opposed to simple), index, access operation, slice, traversal, membership operator, immutability, optional parameters of a function, default values, string module, string format operator.

Related activities:
As stated.

Full-or-part-time: 12h
Laboratory classes: 4h
Self study : 8h

Topic 5. Lists

Description:
This topic in the subject area on the basics of information storage is devoted to lists. Lists are linear data structures par excellence and in Python they are of a native type. As a complement, lists are used to introduce the fundamental concept of mounting type.
Keywords: List and type of list. List of lists. Homogeneous and heterogeneous lists. Operations for accessing the elements in a list. Operations on lists: length, membership, concatenation, repetition. Slices or intervals in lists. The "range" constructor. Mutability: the case of lists. Deleting in lists. Objects, values and aliasing.
Cloning. Iterators in lists. Mutability and parameters: the case of lists. Pure functions and side effects. Matrices. Relation between types and strings.

Related activities:
As stated.

Full-or-part-time: 12h
Laboratory classes: 4h
Self study : 8h

Topic 6. Sequencing schemes

Description:
Unlike the topics covered so far, this topic is methodological. Having introduced the basics of the Python language, this topic deals with iterative design.
The focus is on programming schemes, which are simple and powerful mechanisms in the hands of trained programmers.
Keywords: Programming scheme. Sequence. Traversal scheme. Search scheme. Boolean search. For iterator. Break statement. Else statement (applied to iterations).

Related activities:
As stated.

Full-or-part-time: 6h
Laboratory classes: 2h
Self study : 4h

Topic 7. Modules and files

Description:
This topic is devoted to two areas that are extremely important despite not being connected to the main part of the course: first, modules, which are mechanisms for improving the way code is organised and increasing its reuse, and files, the basic media for storing external information.
Keywords: Module. Import statement. Namespace. Identifier conflict. Continue statement.
Attributes and operators for accessing attributes. Object methods. String methods. List methods. Files.
Text files. Open, read/write, close. End-of-file. Text files. Directories. The sys module. Argv and passing parameters to executable.

Related activities:
As stated.

Full-or-part-time: 12h
Laboratory classes: 4h
Self study : 8h

Topic 8. Tuples

Description:
This intermediate topic does not cover much new ground but does provide an opportunity for thorough revision of the concepts of mutability/immutability and functions. The new concepts are structures called tuples (which must not be confused with registers in Python) and comprehensions, a very powerful tool for working with lists.
Keywords: Tuple (in Python). Operations on tuples. Immutability of tuples. Assignment of tuples: extension to the case of functions. Comprehensions.

Related activities:
As stated.

Full-or-part-time: 6h
Laboratory classes: 2h
Self study : 4h

Topic 9. Dictionaries

Description:
This topic concludes the series of topics on structural data types in Python. Dictionaries are associative structures that enable key-value correspondences to be implemented in a simple manner. They are a very powerful tool. The set of predefined types in Python (strings, lists, tuples and dictionaries) is one of its most esteemed features.
Keywords: Dictionary. Correspondence. Key. Value associated with a key. Insertion and deletion of elements.
Dictionary constructors. Sets of keys and multisets of values

Related activities:
As stated

Full-or-part-time: 20h
Laboratory classes: 4h
Self study : 8h
Self study : 8h

ACTIVITIES

ACTIVITY 1: LECTURE

Material:
Lectures are face-to-face classes that are specifically designed for students' comprehension of the content of the subject. The student participation rate tends to be low.
Support materials:
The support materials are:
- The subject's principle reference work (a textbook that is available online).
- The compulsory reading list.
- The subject's collection of problems.

Full-or-part-time: 12h
Theory classes: 12h

ACTIVITY 2: PROBLEM-SOLVING CLASSES

Description:
These face-to-face classes are devoted to problem solving. They take place in an ordinary classroom and they complement the activity in the laboratory. They require student participation.

Material:
Support materials:
- The subject's principle reference work (a textbook that is available online).
- The compulsory reading list.
- The subject's collection of problems.

Full-or-part-time: 12h
Theory classes: 12h

ACTIVITY 3: LABORATORY CLASS

Description:
Students are asked to do short exercises that complement and aid their comprehension of the subject's topics. The exercises are done in the laboratory and involve implementing software in a computer and its verification. Exercises may have to be finished during independent study.

Material:
The support materials are:
- The subject's principle reference work (a textbook that is available online).
- The subject's collection of problems.
- Manuals for the software used.
- The compulsory reading list

Delivery:
Students are regularly asked to submit a short individual exercise that helps their progress in this activity to be assessed. These exercises must be done in class. They count towards the A value of the final mark.

Full-or-part-time: 42h
Laboratory classes: 27h
Self study: 15h

ACTIVITY 4: INDEPENDENT STUDY

Description:
Independent study may take place individually or in a group. The aim is for students to understand and adopt the skills, vocabulary and techniques that are part of the subject's content

Material:
The support materials are:
- The subject's principle reference work (a textbook that is available online).
- The subject's collection of problems.

Full-or-part-time: 20h
Self study: 20h

ACTIVITY 5: EXERCISES

Description:
Students carry out these exercises independently. The exercises involve solving programming problems, generally without the need for a computer.

Material:
The support materials are:
- The subject's principle reference work (a textbook that is available online).
- The subject's collection of problems

Delivery:
The activity involves solving and handing in several problems that are duly corrected and that are subject to assessment. They count towards the A value of the final mark.

Full-or-part-time: 25h
Self study: 25h

ACTIVITY 6: PROJECT

Description:
Students are required to work on a medium-sized programming project, which consists in implementing and testing a program whose design is described in the project statement. This a group activity that also requires a technical report to be written on the program. It is designed to bring together all of the topics covered in the subject.

Material:
The support materials are:
- The computer laboratory service of the CCEPSEM.
- The project statement and script.
- A sample report.
- Personal class and lecture notes and the rest of the support materials for the subject.

Delivery:
1. The report on the project.
2. The source code for the project.
The assignment must be submitted in the presence of all the members of the work group. The report and the result of the project are assessed. The result of this assessment is the P value of the final mark.

Full-or-part-time: 28h
Theory classes: 4h
Laboratory classes: 4h
Self study: 20h

ACTIVITY 7: EXAM

Description:
The subject has a final examination comprising a set of exercises that must be solved individually on paper without any kind of support material and in a set amount of time.
This activity contemplates the time needed to prepare for the exam.

Material:
Individual answers to the exam questions are handed in and assessed. The result is the F value of the final mark.

Full-or-part-time: 12h
Theory classes: 2h
Self study: 10h

GRADING SYSTEM

The following three items count towards the final mark:
1. The assessment of the student's independent learning (A). This takes into account progress on both theoretical and practical aspects and is based on the compulsory exercises handed in throughout the course. It is divided into A1, the mark for laboratory exercises, and A2, the average mark for the three assignments in the lectures and the mark for a long supervised exercise.
2. The assessment of the project (P). This is based on the delivery in person of the project that may include a public presentation and a report.
3. The assessment of the final exam (F). This is the assessment of the final exam, which brings together all the knowledge and skills acquired during the course.
The final mark is calculated from the items above, which are weighted in the following manner:
Final mark = 0:35A (0:1A1+0:25A2)+ 0:25P + 0:40F

EXAMINATION RULES.

The activities must be carried out according to academic standards and following the rules below:
1. Activities explicitly declared to be individual, whether they are done on site or not, must be done with no contribution from others.
2. Students are obliged to comply with the deadlines, formats and other conditions for submission that are established.
3. The computer laboratory must be used exclusively for academic work. Excessive use of the facility must be avoided.

BIBLIOGRAPHY

Basic:
- Downey, A. Python for software design: how to think like a computer scientist. Cambridge: Cambridge University Press, 2009. ISBN 9780521725965.
- Pilgrim, M. Dive into Python [on line]. New York: Apress, 2004 [Consultation: 30/06/2017]. Available on: http://www.diveintopython3.net/. ISBN 1590593561.
- Guzdial, M.; Ericson, B. Introduction to computing & programming in Python: a multimedia approach. 2nd ed. Upper Saddle River: Pearson/ Prentice Hall, 2010. ISBN 9780136060239.