

Introduction to Programming

Code: 104383
ECTS Credits: 6

Degree	Type	Year	Semester
2503740 Computational Mathematics and Data Analytics	FB	1	1

The proposed teaching and assessment methodology that appear in the guide may be subject to changes as a result of the restrictions to face-to-face class attendance imposed by the health authorities.

Contact

Name: Francesc Xavier Roca Marva
Email: Xavier.Roca@uab.cat

Use of Languages

Principal working language: catalan (cat)
Some groups entirely in English: No
Some groups entirely in Catalan: Yes
Some groups entirely in Spanish: No

Other comments on languages

In the case of Erasmus students or from outside Catalonia, additional classes will be given in the common language between student and teacher so that the student can follow the subject without linguistic problems

Prerequisites

Since the subject is an introduction it will be assumed that there is no prior knowledge about the subject. It is the responsibility of the subject to provide students with a means to acquire the knowledge described in the content section of the subject.

However, it is advisable:

- having attended the technological or scientific baccalaureate,
- Have user-level knowledge of some type of platform (windows, mac or linux)

Objectives and Contextualisation

This subject has a general and introductory character in the programming. It will be deepened in the study of the methodological aspects of programming and the learning of a high level language. Therefore, the general objectives that are proposed for the subject are the following:

- Understand the software life cycle: analyze the problem (understand what is being asked), design (propose a solution to the problem), implementation (coding in a programming language of the chosen solution), test (completion of a test systematically to ensure the correctness of the implemented solution).
- Provide the student with the ability to design algorithms for solving problems, introducing a rigorous and structured programming methodology progressively and systematically, based essentially on the technique of downstream design of algorithms.
- Introduce the student in a real programming language. It is intended that the student perceives the difference between the flexibility of pseudo-algorithmic notation that is used in the first topics and the strict syntax of a real programming language, both in its lexical aspects (valid words of the language), syntactic (rules to combine) and semantic (meaning of the same).

- To habit the student to develop programs following a style norms that seek to achieve quality programs. These rules of style include those that facilitate the compression of the code, such as the use of comments, indentation of the code, the use of names suitable for data types, etc.

Competences

- Apply basic knowledge on the structure, use and programming of computers, operating systems and computer programs to solve problems in different areas.
- Design, develop and evaluate efficient algorithmic solutions to computational problems in accordance with the established requirements.
- Make effective use of bibliographical resources and electronic resources to obtain information.
- Students must be capable of applying their knowledge to their work or vocation in a professional way and they should have building arguments and problem resolution skills within their area of study.
- Students must be capable of communicating information, ideas, problems and solutions to both specialised and non-specialised audiences.
- Students must have and understand knowledge of an area of study built on the basis of general secondary education, and while it relies on some advanced textbooks it also includes some aspects coming from the forefront of its field of study.
- Using criteria of quality, critically evaluate the work carried out.
- Work cooperatively in a multidisciplinary context assuming and respecting the role of the different members of the team.

Learning Outcomes

1. Evaluate and analyse the complexity of computational algorithmic solutions in order to develop and implement that which guarantees best performance.
2. Make effective use of bibliographical resources and electronic resources to obtain information.
3. Select and use algorithmic structures and the representation of appropriate data to solve a problem.
4. Students must be capable of applying their knowledge to their work or vocation in a professional way and they should have building arguments and problem resolution skills within their area of study.
5. Students must be capable of communicating information, ideas, problems and solutions to both specialised and non-specialised audiences.
6. Students must have and understand knowledge of an area of study built on the basis of general secondary education, and while it relies on some advanced textbooks it also includes some aspects coming from the forefront of its field of study.
7. Understand the basic concepts in the structure and programming of computers.
8. Using criteria of quality, critically evaluate the work carried out.
9. Verify and ensure the correct operation of an algorithmic solution in accordance with the requirements of the problem to be resolved.
10. Work cooperatively in a multidisciplinary context, taking on and respecting the role of the distinct members in the team.

Content

Topic 1: Introduction to computer science

History. Functional structure of the computer. Programs / instructions. Conceptual levels of the computer.

Topic 2: Problem solving: introduction to algorithmics and programming.

Introduction to problem solving. Algorithm concept Phases in the development of algorithms. Programming as an engineering discipline. Software life cycle. Basic elements of an algorithm. Tools for the representation of algorithms. Programming languages. Classification. Language translators: Compilers and interpreters.

Topic 3: Basic concepts and control structures

Definition of variables and constants. Type of fundamental data. Sequential structure. Selection or conditional structures. Iterative or repetitive structures.

Topic 4: Subprograms

The concept of a subprogram as an abstraction of operations. Location, neously, scope and visibility. Definition of functions and procedures. Calls to functions and procedures. Modular design descending.

Topic 5: Error control

Types of errors Exceptions and assertions. Preventive programming Debugging programs.

Topic 6: Introduction to object-oriented programming

Classes and objects. Attributes and methods. Encapsulation Definition of classes.

Topic 7: Complex data types

Arrays (one-dimensional and multidimensional). Lists, tuples and sets. Dictionaries

Topic 8: Archives

Basic definitions. Input / output of data in files. Type of access to files.

Methodology

The teaching management of the subject will be done through the documentary manager Caronte (<http://caronte.uab.cat/>), which will be used to view the materials, manage the practices groups, make the corresponding deliveries, see the notes, communicate with the teachers, etc. To be able to use it you have to do the following steps:

1. Register as a user giving the name, NIU, and a photo ID in JPG format. If you have already registered for another subject, you do not need to do it again, you can go to the next step.
2. Register for the type of teaching "Introduction to Programming", giving as a subject code "IniProg" (without quotes).

In the development of the subject, five types of teaching activities can be differentiated:

MD1 Exposition of contents in class - Master class: The typical structure of a master class of this type will be the following: in the first place, one will be done Introduction where the objectives of the exhibition and the contents to be discussed will be briefly presented. In order to provide the appropriate context, the presentation will refer to the material exhibited in previous classes, so that the position of these contents is clarified within the general framework of the subject.

MD2 Participatory classes: Joint resolution of the set of problems proposed to students. All the subjects will be accompanied by a relation of problems that the student must try to solve. In this sense, and as the student progresses in the depth of their knowledge, these problems will be gradually more complex, allowing in this way to clearly appreciate the advantages of using the methodological tools taught during the course. All the problems developed in class and others that may be proposed can be found on the Caronte platform, and will be self-assessed. These activities should allow students to deepen their understanding and personalize knowledge. The fact that they are self-assessed allows you to adjust the pace of consolidation and reflect on your own learning.

MD3 Tutorials: Hours of free disposal for the student for inquiries about aspects that need additional help from the faculty.

MD4 Carry out short projects: Completion of practices, larger problems, to deepen in applied aspects of the theory. The practical part of each subject will be completed with at least one practical session, where students will have to solve specific problems of a certain complexity. These projects will be solved in small groups. Each member of the group will have to do a part and put it in common with the rest to have the final solution.

MD5 Assessment activities: See the teaching guide's assessment section

Cross-curricular

Transversal competences T01, T02 and T04 are worked and evaluated at the home of the course in the following activities.

Competition	MD1	MD2	MD4
T01 - Evaluate critically and with quality criteria the work carried out	Personal work for the preparation of materials Assessed: Exam Theoretician (Tp and Tf)	Completion of the proposed problems Evaluated: Resolution Problems (P) of self-evaluation with the evaluate	Evaluation: Laboratory practice (PLg) Group Materials will be provided on how to work effectively in a team. Assessed: Practical Laboratory (PLg) Group
T02 - Work cooperatively in a multidisciplinary context assuming and respecting the role of the different team members			
T04 - Effective use of the bibliography and electronic resources to obtain information	Personal work for the preparation of materials Assessed: Theoretical Exam (Tp and Tf)		Work: Find information to solve the practice Evaluated: Laboratory PLg and PLi practice

Activities

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Practical classes	10	0.4	8, 1, 7, 5, 3, 10, 2, 9
Presential Classes/ Theory	10	0.4	8, 1, 7, 6, 5, 4, 3, 2, 9
Problem classes	26	1.04	8, 1, 7, 4, 3, 9
Type: Supervised			
Proyecto de programación	17.5	0.7	8, 1, 7, 5, 4, 3, 10, 2, 9
Type: Autonomous			
Preparation classes and personal study	20	0.8	8, 1, 7, 6, 5, 4, 3, 2, 9
Resolution of practical cases (group).	22.5	0.9	8, 1, 7, 4, 3, 10, 2, 9
Resolution of self-assessed problems (individual)	40	1.6	8, 1, 7, 4, 3, 2, 9

Assessment

Processes and scheduled evaluation activities

The following assessable activities are carried out:

Activity	Date	Recovery	Percentage Block	Minimum note
Theoretical Partial Exam (Tp) Individual	Consult Planning	Consultation Planning	15%	No
Final Theoretical Examination (Tf) Individual	Consult Planning	Consultation Planning	35%	Tf> = 5
Resolution Problems (P) Individual	Each Week	No	20%	No
Laboratory Practice (PLg) Group	Consult Planning	Consultation Planning	20%	PLg> = 5
Laboratory Practices (PLi) individual	Check Planning	Consult Planning	10%	PL>> 5

Note of each final of the subject is the weighted sum of the assessment activities with their corresponding percentage. The result of the weighted sum must be ≥ 5 to consider the course approved.

Programming of evaluation activities

The dates of evaluation and delivery of works will be published in the documentary manager Caronte and may be subject to changes of programming for reasons of adaptation to possible incidents. The document manager will always be informed about these changes since it is understood that this is the usual platform for exchanging information between teachers and students.

Recovery procedure

The student can submit to the recovery whenever it has been presented to a set of activities that represents a minimum of two thirds of the total grade of the subject.

The theoretical examinations (Tp and Tf) can be recovered in a single examination of recovery to the dates fixed by the coordination of the Degree. The recovery exam will have a percentage of 50% on the final grade. The exam of the examination of recovery, in case of realization, will replace the notes of the partial exams (Tp) and final (Tf) realized during the course.

In accordance with the coordination of the Degree and the direction of the School of Engineering, the Individual Problem Solution (P) activity can not be recovered.

Procedure for reviewing the qualifications

Students will have the right to review the theoretical exams (Tp and Tf). The site, date and time of review will be published on the day the notes are published. If the student does not submit to this review, this activity will not be reviewed later. Only in justified cases can a posteriori review of the fixed date and always up to a maximum of 7 calendar days.

Of the notes of the laboratory practice (PLg and PLi) there will be no review because the assessment is carried out in front of the students.

Qualifications

Non-Appraising(NA): Any student who submits a practice or a scheduled assessment will note. It will only be deemed not evaluable in the case of not giving any evaluable activity.

Final Note: It is based on the weighted sum according to the criteria set out in the evaluation activities section. If in any activity the minimum grade is not obtained, the note will come out of the following formula:

Minimum (apply a weighted sum and 4.9)

Honorary enrollments

Granting a grade of honor registration is the decision of the faculty responsible for the subject. The regulations of the UAB indicate that MH can only be granted to students who have obtained a final grade equal to or greater than 9.00. You can grant up to 5% of MH of the total number of students enrolled. In case the number of students with a grade greater than or equal to 9 is more than 5% of the total number of students enrolled, students were prioritized according to the following rules (in order):

1. Students who have made fewer recoveries of activities (either because they have suspended the activity or because they have gone up to grade).
2. Students with more grades than 9 to activities 1, 2, and 5.
3. Students with a better overall score.

Irregularities by the student, copy and plagiarism

Without prejudice to other disciplinary measures that they deem appropriate, the irregularities committed by the student that could lead to a variation of the grade of an evaluation act will be scored with a zero. Therefore, copying, plagiarism, cheating, letting copy, etc. in any of the evaluation activities will involve suspending with a zero. The evaluation activities qualified in this way and by this procedure will not be recoverable. If it is necessary to pass any of these evaluation activities to pass the subject, this subject will be suspended directly, without the opportunity to recover it in the same course. In this case, the numerical note of the file will be the lower value between 3.0 and the weighted average of the notes.

Evaluation of the repeating studentsrepeating

The students must complete the course as a whole, there is no note from previous courses.

Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes
Final Theoretical Exam (Tf) Individual	35%	2	0.08	8, 1, 7, 6, 5, 4, 3, 2, 9
Laboratory Practices (PLg) in Group	20%	0.5	0.02	8, 1, 7, 5, 4, 3, 10, 9
Laboratory Practices (PLi) individual	10%	0.5	0.02	8, 1, 7, 5, 3, 2, 9
Partial Theoretical Exam (Tp) Individual	15%	1	0.04	8, 1, 7, 6, 5, 4, 3, 2, 9
Resolution Problems (P) Individual	20%	0	0	8, 1, 7, 4, 3, 2, 9

Bibliography

J. Guttag. *Introduction to Computation and Programming Using Python: With Application to Understanding Data Second Edition*. MIT Press. ISBN-10: 9780262529624

S. Chazallet *Python 3. Los fundamentos del lenguaje*. Eni, ISBN-10: 2409006140

E. Matthes. *Python Crash Course: A Hands-On, Project-Based Introduction to Programming*. No Starch Press ISBN-10: 1593276036

M. Myers. *A Smarter Way to Learn Python: Learn it faster. Remember it longer*. Createspace Independent Pub ISBN-10: 1974431479

A. Prieto, A. Lloris, J.C. Torres. *Introducción a la Informática*. Mc Graw-Hill ISBN-10: 8448146247

A. Prieto, B. Prieto. *Conceptos de Informática*. Mc Graw-Hill, Schaum ISBN-10: 8448198573

L. Joyanes Aguilar. *Fundamentos de Programación: Algoritmos, Estructuras de Datos y Objetos*. Mc. Graw-Hill. ISBN-10:8448161114