UAB
Universitat Autònoma de Barcelona

Foundations of Programming

Code: 104337
ECTS Credits: 6

2025/2026

Degree	Type	Year
Data Engineering	FB	1

Contact

Name: Roberto Benavente Vidal

Email: robert.benavente@uab.cat

Teachers

Ivan Erill Sagales

Teaching groups languages

You can view this information at the end of this document.

Prerequisites

Given that this is an introductory course, it is assumed that students do not have any prior knowledge of the subject.

However, it is advisable to have a basic knowledge of any platform (windows, mac or linux).

In face-to-face activities in the classroom, it is essential to have a laptop to carry out the assignments of the subject.

Objectives and Contextualisation

This subject has a general and introductory character to programming. The study of the methodological aspects of programming and the learning of a high level language will be deepened. Therefore, the proposed general objectives are as follows:

- Understanding the life cycle of the software: analysis of the problem (understanding what is being asked), design (proposing a solution to the problem), implementation (coding in a programming language the chosen solution), and test (carrying out a test in a systematic way to ensure the correctness of the implemented solution).
- Providing the student with the ability to design algorithms for problem solving, introducing a rigorous and structured programming methodology in a progressive and systematic way (based on the technique of descending algorithm design).
- Introducing the student to a real programming language. It is intended that the student perceives the difference between the flexibility of the pseudo-algorithmic notation used in the first topics and the strict

syntax of a real programming language in different aspects: lexical (valid words of the language), syntactic (rules to combine them) and semantic (meaning of them).
- Making the student familiar with developing programs following some norms of style tending to obtain high quality programs. These style rules include those that facilitate the comprehension of the code, such as the use of comments, the indentation of the code, the use of appropriate names for data, etc.

Competences

- Design efficient algorithmic solutions to computational problems, implement them in the form of robust software developments which are structured and easy to maintain, and verify their validity.
- Make a critical evaluation of work carried out.
- Plan and manage the available time and resources.
- 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.

Learning Outcomes

1. Apply strategies for debugging, testing and correcting programmes.
2. Apply the basic principles of computer structure and programming.
3. Develop programmes that are well documented, using a good programming style.
4. Implement medium-difficulty algorithmic problems in a programming language.
5. Make a critical evaluation of work carried out.
6. Plan and manage the available time and resources.
7. 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.

Content

Unit 1: Introduction to the computer

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

Unit 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.

Unit 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.

Unit 4: Data structures

Unidimensional arrays: strings, tuples and lists. Search and traverse.

Unit 5: Subprograms

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

Unit 6: Files

Basic Definitions. Input / output of data in files. Types of access to files.

Unit 7: Prevention and detection of errors

Types of errors. Exceptions. Debugging programs.

Unit 8: Introduction to object-oriented programming

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

Unit 9: Complex data structures

Lists: iterators, generators, functional paradigm, and list comprehensions. Sets. Dictionaries.

Activities and Methodology

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Practical Labs	10	0.4	2, 1, 5, 3, 4, 6, 7
Problems classes	30	1.2	2, 1, 5, 3, 4, 7
Theory classes	10	0.4	2, 1, 5, 3, 4, 7
Type: Supervised			
Programming projects	25	1	2, 1, 5, 3, 4, 6, 7
Type: Autonomous			
Classes preparation and personal work	30	1.2	2, 1, 3, 4, 6, 7
Solving self-assessment problems	41	1.64	2, 1, 5, 3, 4, 7

The management of the teaching of the subject will be done through the documentary manager Caronte (http://caronte.uab.cat/), which will serve to see the materials, manage the groups of practices, make the corresponding deliveries, see the marks, and communicate with the teaching staff, etc. In order to use it, it is necessary to do the following steps:

1. Register by giving the name, NIU, and a photo 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. Enroll in the type of teaching "Programació en Python", giving as a subject code the one provided on the first day of class.

The course will be developed using a flipped classroom methodology. This means that students will need to develop some knowledge before the in-person sessions, which will primarily involve practical activities aimed at addressing as many cases as possible to understand the complexity involved in solving a programming problem.

Before the class. The basic concepts are covered before the in-person sessions using materials provided a week in advance on the Caronte platform. The materials covered will include written documentation, multimedia content, and learning activities.

Each student should prepare for classes before attending, reviewing the materials as many times as necessary to assimilate the concepts. As a guide, it is recommended to dedicate a number of hours per week equivalent to the class time, approximately 2 to 3 hours per week. For each topic, a questionnaire will be provided to assess the degree of achievement of the content covered.

The class. The objective of the in-person sessions is to move from information to knowledge. We will emphasize creating knowledge applicable to the analysis and solution of specific problems. The class structure will be:

Review of the basic concepts of thetopic to be covered (15 min). The objective is to resolve any doubts that may exist regarding theoretical aspects that are not understood and to monitor the acquisition of basic concepts.

Problem-solving (remainder of class time). The goal is to experiment with a practical task that allows you to apply the concepts learned to solve challenges and thus share and create knowledge.

After the class. The goal is to consolidate knowledge. The Caronte platform offers a set of assessable, self-correcting problems that should allow students to deepen their understanding of the topics and personalize their knowledge. The self-assessment allows students to adjust the pace of consolidation to each student and foster reflection on their own learning.

Annotation: Within the schedule set by the centre or degree programme, 15 minutes of one class will be reserved for students to evaluate their lecturers and their courses or modules through questionnaires.

Assessment

Continous Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes
Final exam	45%	2	0.08	2, 1, 5, 3, 4, 7
Graded in-class activities	10%	0	0	2, 1, 5, 3, 4
Intra-semester exam	15%	2	0.08	2, 1, 5, 3, 4, 7
Practical works	15%	0	0	2, 1, 3, 4, 6
Problem-solving assignments	15%	0	0	2, 1, 3, 4

Scheduled evaluation activities

The subject consists of the following assessment activities:

Activity	Type	Weight	Minimum grade	Recoverable
Intrasemester exam	individual	15%	No	Yes

Final exam	individual	45%	5	Yes
Lab works	group	15%	5	Yes
Problem-solving assignments	individual	15%	No	Yes
Graded class activities	individual	10%	No	No

To pass the course, you must have a minimum grade of 5 on the final exam and the Lab works.

Scheduling evaluation activities

The schedule of assessment activities will be announced on the first day of the course and will be made public through Caronte and in the exams section of the degree website.

Recovery process

Students may take the make-up exam provided they have completed a set of activities that represents at least two-thirds of the total grade for the subject. In particular, if the student has not taken the final exam, they will not be able to take the make-up exam.

The final exam may be made up on the date set by the degree coordinator. If a make-up exam is taken, the grade for this exam may also replace the intra-semester exam grade if the make-up exam grade is higher than the intra-semester exam grade.

The practical Lab assignments can be made up in a second submission on the date specified in the assessment activities calendar, which will be given on the first day of the course and made public through Caronte.

Problem submissions can be made up at any time during the course by submitting them again and evaluating them with the auto-corrector.

According to the Degree Coordinator and the School of Engineering administration, evaluable class activities cannot be recovered.

Qualification review procedure

For exams (mid-semester, final, and make-up exams), a review location, date, and time will be indicated so students can review the activity with the faculty. In this context, students may submit complaints about the activity grade, which will be evaluated by the faculty responsible for the subject. If the student does not attend this review, the activity will not be reviewed later.

For all other activities, if any errors are detected in the correction, you may request a review at any time, up until the date of the make-up exam review.

Ratings

Non-Evaluable (NA): A student will be considered non-evaluable (NA) if he/she has not attended any of the following evaluation activities: subject exams (intra-semester, final and recovery), practicals and evaluable activities in class.

Final grade: Calculated as the weighted sum of the evaluation activities according to the criteria set out in the evaluation activities section.

If a student fails the course because one of the assessment activities does not achieve the minimum required grade, the numerical grade on the transcript will be the lower of 4.5 and the weighted average of the grades. The exceptions are that a grade of "Not Assessable" will be awarded to students who do not participate in any of the assessment activities, and the numerical grade on the transcript will be the lower between 3.0 and the weighted average of the grades if the student has committed irregularities in an assessment activity.

Honors grade: Awarding an honor roll is the decision of the faculty responsible for the course. UAB regulations state that honor rolls may only be awarded to students who have obtained a final grade equal to or higher than 9.00. Honor rolls may be awarded to up to 5% of the total enrolled students. If the number of students with a grade greater than or equal to 9 exceeds 5% of the total enrolled, the following prioritization criteria will be applied in the order listed below:

1. Students who have completed fewer recovery activities.
2. Students with the highest final grade.
3. Students with the highest grade on the final exam
4. Students with the highest grade on the mid-semester exam

Irregularities committed by students

Without prejudice to other disciplinary measures deemed appropriate, and in accordance with current academic regulations, any irregularities committed by a student that may lead to a grade variation in an assessable activity will be graded with a zero (0). Assessment activities graded in this manner and by this procedure will not be recoverable. If it is necessary to pass any of these assessment activities to pass the course, the subject will be immediately failed, with no opportunity to recover it in the same academic year. These irregularities include, among others:

- the total or partial copying of a practice, report, or any other assessment activity;
- allow copying;
- present group work not entirely completed by the group members (applied to all members, not just those who have not worked);
- unauthorized use of AI (e.g., Copilot, ChatGPT or equivalent) to solve exercises, practices and/or any other evaluable activity;
- present as one's own materials prepared by a third party, even if they are translations or adaptations, and generally works with non-original and exclusive elements of the student;
- have communication devices (such as mobile phones, smart watches, camera pens, etc.) accessible during individual theoretical-practical assessment tests (exams);
- talk with classmates during individual theoretical-practical assessment tests (exams);
- copying or attempting to copy from other students during theoretical-practical assessment tests (exams);
- use or attempt to use material relatedto the subject during theoretical and practical assessment tests (exams), when these have not been explicitly permitted.

In future years of this course, students who have committed irregularities in an assessment process will not have any of their completed assessment activities validated, nor will they be eligible for any specific assessment regulations for repeating students.

Uses of AI

For this subject, the use of Artificial Intelligence (AI) technologies is permitted exclusively for support tasks, such as searching for information, interpreting and explaining programs, answering questions, etc. Under no circumstances is AI permitted for programming assignments submitted as part of the assessment. Any work that includes AI-generated fragments will be considered a breach of academic honesty and will result in the application of the regulations regarding irregularities by the student.

Evaluation of repeating students

If the student successfully completed the "Advanced Programming" course in the previous or current academic year and passed the review exercises for that course, he or she may opt for an assessment consisting of a summary test, without having to complete all the assessment activities. The final grade in this case will be 5.

To be eligible for this differentiated assessment, the interested person must request it from the teacher by email (Robert.Benavente@uab.cat or Ivan.Erill@uab.cat) no later than 15 days after the start of classes.

Single evaluation

This subject offers a single evaluation system.

The single assessment of the subject will consist of the following assessment activities:

Activity	Type	Weight	Minimum grade	Recoverable
Final exam	individual	60%	5	Yes
Lab works	group	15%	5	Yes
Problem-solving assignments	individual	15%	No	Yes
Practical Questionnaire	individual	10%	5	Yes

To pass the course, you must achieve a minimum grade of 5 on the final exam,thepractical lab works, and the practical questionnaire.

The same make-up system will be applied as for continuous assessment, except for the items referring to the intra-semester exam. The practical questionnaire can be made up on the day of the make-up exam.

The review of the final grade follows the same procedure as for continuous assessment.

The same non-assessable criterion will be applied as for continuous assessment.

Bibliography

J. L. Gahete Díaz, C. M. Vallez Fernández, A. Fernández-Pacheco Sánchez-Migallóm. *Fundamentos de programación con Python*. McGraw Hill, 1ª edición, 2024. ISBN-10: 9788448645274

J. Guttag. *Introduction to Computation and Programming Using Python: With Application to Understanding Data*. MIT Press, 2nd edition, 2016. ISBN-10: 9780262529624

S. Chazallet. *Python 3: Los fundamentos del lenguaje*. Ediciones ENI, 2ª edición, 2016. ISBN-10: 2409006140

E. Matthes. *Python Crash Course: A Hands-On, Project-Based Introduction to Programming*. No Starch Press, 1st edition, 2016. ISBN-10: 1593276036 (Online: https://ebookcentral.proquest.com/lib/uab/detail.action?docID=4503145)

M. Myers. *A Smarter Way to Learn Python: Learn it faster. Remember it longer*. Createspace Independent Pub, 2017. ISBN-10: 1974431479

L. Joyanes Aguilar. *Fundamentos de Programación: Algoritmos, Estructuras de Datos y Objetos*. 5ª Edición, McGraw-Hill, 2020. ISBN-10:8448161114 (Online:

https://www-ingebook-com.eu1.proxy.openathens.net/ib/NPcd/IB_Escritorio_Visualizar?cod_primaria=1000193&l
)

Software

The latest version of the Anaconda package will be used, which includes Python 3.x and the Spyder editor (https://www.anaconda.com/download/success).

Groups and Languages

Please note that this information is provisional until 30 November 2025. You can check it through this link. To consult the language you will need to enter the CODE of the subject.

Name	Group	Language	Semester	Turn
(PAUL) Classroom practices	81	Catalan	first semester	morning-mixed
(PAUL) Classroom practices	82	Catalan	first semester	morning-mixed