UAB
Universitat Autònoma de Barcelona

Fundamentals of Programming I

Code: 106553
ECTS Credits: 9

2024/2025

Degree	Type	Year
2504392 Artificial Intelligence	FB	1

Contact

Name: Alicia Fornes Bisquerra
Email: alicia.fornes@uab.cat

Teaching groups languages

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

Prerequisites

None

Objectives and Contextualisation

This subject aims to provide a general introduction to programming. The general objectives are:

- Understanding the life cycle of software: analysis of the problem, design, implementation, and test.
- Designing algorithms for problem solving, with a rigorous structured programming methodology.
- Introducing a real programming language, perceiving the difference between a pseudo-algorithmic notation and the syntax of a real programming language. It includes the understanding of the lexical (valid words of the language), syntactic (rules to combine them) and semantic (meaning) aspects.
- Understanding and applying the basic concepts of programming: control structures, data structures and subprograms.
- Understanding and applying the basic principles of object-oriented programming: concept of class, object, data encapsulation, inheritance.

- Introducing the concept of recursion and its application.
- Familiarizing with developing environments, following some norms of style, and rules for a better comprehension of the code: comments, indentation, etc.

Competences

- Analyse and solve problems effectively, generating innovative and creative proposals to achieve objectives.
- Design, implement, analyse and validate efficient and robust algorithmic solutions to computational problems derived from the design of intelligent systems.
- Know and efficiently use techniques and tools for representation, manipulation, analysis and management of large-scale data.
- 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.

- Work cooperatively to achieve common objectives, assuming own responsibility and respecting the role of the different members of the team.

Learning Outcomes

1. Analyse and solve problems effectively, generating innovative and creative proposals to achieve objectives.
2. Apply debugging, testing and correction strategies to check the correct functioning of an algorithmic solution according to the requirements of the problem at hand.
3. Apply object-oriented programming principles to programme development.
4. Develop well-documented programs with a good programming style.
5. Select and use the most appropriate algorithmic and data representation structures in order to solve a problem.
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 common data structures and the algorithms that use and manipulate them.
8. Understand the limitations and advantages of different data representations and how to choose the most efficient structures to facilitate the subsequent data analysis and exploitation.
9. Work cooperatively to achieve common objectives, assuming own responsibility and respecting the role of the different members of the team.

Content

- Problem solving: Introduction to algorithmic and programming
- Basic concepts and control structures
- Data structures.
- Subprograms
- Files
- Prevention and detection of errors
- Introduction to object-oriented programming
- Introduction to recursion

Activities and Methodology

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Problem solving seminaries	50	2	1, 2, 3, 4, 5
Theoretical lessons	25	1	7, 8
Type: Supervised			
Programming projects	50	2	1, 2, 3, 4, 5, 7, 8, 9
Type: Autonomous			
Individual work	35	1.4	1, 2, 3, 5, 7, 8
Problem resolution	60	2.4	1, 3, 4, 5, 7, 8

The platform Caronte (http://caronte.uab.cat/) will be used to share the materials, deliver the works/projects, consult the marks of the subject, communicate with the teaching staff, etc. To enroll the course, you must register (with name, NIU and photo) and enroll de subject (the subject code is provided on the first day of class).

The teaching methodology will be mainly focused on practical work. Classroom sessions will be organized to discuss the theoretical contents of the subject, followed by exercises and programming problems. Concretely, the different types of teaching activities are the following:

- Exhibition of contents. Theory activities are aimed at consolidating the most theoretical aspects of the subject, from a very practical perspective with examples. Some of these theoretical contents must have been prepared before the class: reading texts, viewing videos, searching for information, etc.
- Participative lectures: Joint resolution of the set of proposed problems to consolidate the theoretical contents. All problems will be available in the Caronte platform, and will be self-evaluable. These activities allow the student to deepen understanding and to get personalized knowledge. They are self-evaluable to allow adjusting the pace of consolidation and to reflect on our own learning.
- Projects: Programming short practical projects to deepen the applied theoretical concepts. These projects will be resolved in small groups, where each member must do a part of the work and put it in common with the rest of the group to have the final solution.
- Tutoring sessions will be freely available for students to solve questions regarding the course.

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.

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
1st Partial theory exam	25	2	0.08	1, 3, 4, 5, 7, 8
2nd Partial theory exam	25	2	0.08	1, 2, 3, 4, 5, 7, 8
Delivery of problems	10	0	0	1, 2, 3, 4, 5, 7, 8
Programming projects	40	1	0.04	1, 2, 3, 4, 5, 6, 7, 8, 9

The evaluation will be continuous, so there is no unique evaluation modality.

- Exams (Par): There will be two individual written theoretical-practical exams. The first one (Par1) will be done approximately halfway through the semester. The second one (Par2) will take place at the end of the semester and evaluate the theoretical concepts not evaluated in Par1.
- Recovery exam: In case the Theory Note does not reach the pass (5), students may do a recovery exam on the failed part.
- Problems (NProb): Most weeks there will be the possibility to deliver activities worked during the week. The delivery is through Caronte.
- Projects (Proj): The grade of the projects will be obtained from the weighted average of the grades obtained in the two projects. Each note will be the arithmetic mean of the group note and the individual

note. Each one of the deliveries must be passed separately. In case of failing any delivery, there will be the option to recover the failed part. The grade of the recovery will be a maximum of 5.

GRADE INDICATORS:

The final mark is calculated as follows:

- Partial Score 1 = Par1
- Partial Score 2 = Par2
- Theory Mark = 0'5*Par1 +0'5*Par2
- Project Mark = 0'5*Proj1 + 0'5*Proj2
- COURSE FINAL GRADE = 0'5 * Theory Mark + 0'4 * Project Mark + 0'1 * NProb

EVALUATION CRITERIA

- For considering the Theory Mark, it is necessary to obtain a minimum mark of 4 in each partial to make the average.
- For considering the Project Mark, it is necessary to obtain a minimum mark of 4 in each delivery to make the average.
- If the student fails some of the project deliveries, the maximum score that can be obtained in the recovery is 5.
- To pass each part (theory and project), a minimum of 5 must be obtained.
- The course will be passed if the COURSE FINALSCORE is greater than or equal to 5.
- In case of not reaching the minimum required in any of the evaluation activities, the numerical grade of the record will be the lower value between 4.5 and the weighted average of the grades.

NOT EVALUABLE: If the student does not deliver any evaluation activity.

REPEATING STUDENTS: No separately approved part (theory, project) is kept/validated from one academic year to another.

HONORS (MH): Awarding an honors degree is the decision of the teaching staff responsible for the subject. UAB regulations indicate that MH can only be granted to students who have obtained a final grade equal to or greater than 9.00. Up to 5% MH of the total number of students enrolled can be awarded.

EVALUATION SCHEDULE:

- Partial exams: according to the academic calendar of the School of Engineering.
- Recovery Exam: according to the academic calendar of the School of Engineering.
- Deliveries of activities: date and time set in advance at Caronte.
- Deliveries of the project: date and time fixed in advance at Caronte.

The dates of deliveries at Caronte may be subject to program changes for reasons of adaptation to possible incidents. These changes will always be reported at Caronte, as it is the usual mechanism for the exchange of information between teachers and students.

For each assessment activity, a place, date and time of review will be indicated in which the student can review the activity with the teacher. In this context, claims may be made about the grade of the activity, which will be evaluated by the teachers responsible for the subject. If the student does not come to this review, the activity will not be reviewed later.

COPIES AND PLAGIARISM

Without prejudice to others that are deemed appropriate and in accordance with current legislation academic discipline, irregularities committed by a student that can lead to a variation of the rating will be rated with a zero mark (0). Assessment activities classified in this way and by this procedure will not be recoverable. If it is necessary to pass any of these assessment activities to pass the course, this course will be suspended directly, with no opportunity to recover it in the same course. These irregularities include, among others:

- the total or partial copy of a practice, report, or any other evaluation activity;

- let other copy your exam/work;
- present a group work that has not been entirely done by the members of the group;
- present as own those materials produced by a third party, even if they are translations or adaptations, and in general works with non-original and exclusive elements of the student;
- have communication devices (such as mobile phones, smartwatches, etc.) accessible during the theoretical assessment tests - individual practices (exams).
- In case the student has committed irregularities in any evaluation part (and therefore it will not be possible to pass via compensation), the numerical grade of the subject will be the lower value between 3.5 and the weighted average of the grades. In summary: copying, let others copy your work or plagiarizing in any of the evaluation activities is equivalent to a failure with a grade lower than 3.5.

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.
- D. Phillips, C. Giridhar, S. Kasampalis. Python: master the art of design patterns. Packt Publishing, 2016.
- Steven F. Lott. Mastering object-oriented Python. Packt publishing, 2014.
- Clean code: a handbook of agile software craftmanship. R.C. Martin. Prentice Hall, 2008.

Software

Anaconda, which includes Python and Spyder (https://www.anaconda.com/download)

Language list

Name	Group	Language	Semester	Turn
(PAUL) Classroom practices	71	English	first semester	afternoon