UAB
Universitat Autònoma de Barcelona

Fundamentals of Programming II

Code: 106554
ECTS Credits: 6

2025/2026

Degree	Type	Year
Artificial Intelligence	FB	1

Contact

Name: Javier Panadero Martinez

Email: javier.panadero@uab.cat

Teachers

Haoyuan Li

Teaching groups languages

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

Prerequisites

The course doesn't have any official prerequisites. Nevertheless, it builds on top of Fundamentals of Programming I. Therefore, it is recommended for students to have a good knowledge of the basics of programming and Python.

Objectives and Contextualisation

This course is the continuation of Fundamentals of Programming I, and the general objectives are:

- Understanding different data structures such as stacks, queues, and lists.
- Understanding different more complex data structures such as hash tables, trees, and graphs.
- Understanding dynamic memory and pointers.
- Understanding different algorithms for sorting and searching, and the limitations of those algorithms.
- Be able to analyze a complex problem, design an efficient solution, implement it, calculate its cost and test it.
- Introduce the concept of recursion and its application, as well as being able to analyze the complexity of recursive algorithms.

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.
- Develop critical thinking to analyse alternatives and proposals, both one's own and those of others, in a well-founded and argued manner.
- Introduce changes to methods and processes in the field of knowledge in order to provide innovative responses to society's needs and demands.
- Know and efficiently use techniques and tools for representation, manipulation, analysis and management of large-scale data.

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 critical thinking to analyse alternatives and proposals, both one's own and those of others, in a well-founded and argued manner.
5. Develop well-documented programs with a good programming style.
6. Evaluate and analyse the computational complexity of algorithmic solutions in order to develop and implement the one that guarantees the best performance.
7. Implement recursive solutions to programming problems.
8. Propose new methods or informed alternative solutions.
9. Select and use the most appropriate algorithmic and data representation structures in order to solve a problem.
10. Understand common data structures and the algorithms that use and manipulate them.
11. 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.

Content

Dynamic data structures:

- Pointers, dynamic data structures (lists, stacks, queues).
- Non-linear data structures. Hashing techniques. Hash Matrices and Hash lists.
- Recursion.
- Sort algorithms.
- Search algorithms.
- Graphs.
- Trees.
- Analysis of the complexity of algorithms.

Advanced programming of different dynamic data structures in C language.
Basic programming of different dynamic data structures in C++ language.
Import of C code from Python.
Programming different Python data structures in C.
Programming languages:

- Structured language C (basic and advanced)
- Object-oriented language C++ (basic)
- Python interpreted language (knowledge of Fundamentals of Programming I)

Activities and Methodology

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Basic exercises realization	5	0.2	3, 2, 10, 7, 9
Practical projects	12	0.48	1, 3, 2, 6, 10, 11, 4, 5, 7, 8, 9
Problem solving	16	0.64	3, 2, 10, 4, 7, 9
Theory	10	0.4	3, 10, 11, 7, 9
Type: Autonomous			
Autonomous study	56	2.24	1, 3, 2, 6, 10, 11, 4, 5, 7, 8, 9
Practical projects preparation	30	1.2	1, 3, 2, 6, 10, 11, 4, 5, 7, 8, 9
Preparation of problems and basic exercises	10	0.4	3, 2, 6, 10, 11, 7, 9

We will use the moodle platform (Caronte or Virtual Campus) to share materials, deliver projects and other administrative activities. All the details will be published at the beginning of the course.

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. More specifically, the different types of teaching activities are the following:

- Theoretical classes: Explanation on the board of the theoretical part of each topic of the program. The typical structure of a class will be the following: in the first place an introduction will be made where the objectives of the class and the contents to be discussed will be briefly presented. Next, the contents of a topic will be covered, including narrative expositions, formal developments that provide theoretical foundations, and intercalating practical examples, which illustrate the application of the exposed contents. Finally, the professor will present the conclusions of the discussed contents. The theoretical classes, in spite of being mainly an explanation by the teacher, will be participatory for the students giving them the opportunity to ask those points that they do not finish understanding. The students will be constantly asked questions and problems to check the good understanding of the exposed subject. There will be 2 exams to evaluate this part of the subject.

- Problem-solving classes: Joint resolution of problems. All the topics will be accompanied by a set of problems that the student must try to solve. In this sense, and as the student progresses in and deepens in their knowledge, these problems will be gradually more complex. The classes will be the natural forum in which the development of practical work can be discussed in common, contributing the knowledge that the student is lacking tocarry it forward. The mission of the problem-solving classes is to bridge between theoretical classes and classes in the laboratory, which will promote the ability to analyze and synthesise, the development of critical reasoning, and that will train the student in the resolution of problems. During the course, students (in groups of 2 persons) will do various short and basic exercises that will be evaluated. Each group will solve these basic exercises and make the delivery within the corresponding short period. This exercise seeks to enable students to solve theoretically the concepts worked in theoretical classes and then be able to apply them to the practical laboratory case.

- Practical projects: Students will work in pairs, and go through the different steps of the software development life cycle to implement the solution to a problem. This practical projects will complete the theoretical subjects, where the student will develop several programs and will have to try to solve a specific proposed problem. These projects will have to be delivered at the indicated dates. There will be 2 practical projects.

This approach to work is aimed at promoting active learning and developing competencies of organizational and planning skills, oral and written communication, teamwork and critical reasoning. The quality of the exercises carried out, of its presentation and of its operation will be valued especially.

All the details of the subject, the calendar and the evaluation dates will be informed at the beginning of the course and will appear on the communication platform.

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
Basic exercises	20%	5	0.2	3, 2, 6, 10, 11, 7, 9
Partial individual exam 1	25%	2	0.08	6, 10, 9
Partial individual exam 2	25%	2	0.08	3, 6, 10, 7, 9
Practical Projects	30%	2	0.08	1, 3, 2, 6, 10, 11, 4, 5, 7, 8, 9

The objective of the assessment process is to verify that the student has achieved the knowledge and skills defined in the objectives of the subject, as well as the associated competences.

Three types of activities will be assessed independently, and the weighted sum of them will give the final grade. These three activities are:

1. Theory (T)
2. Solution of the practical projects (PP)
3. Resolution of basic exercises (EXER).

The Theory (T) part will be assessed with two individual partial exams throughout the course. The final grade of Theory will come from the weighted sum of the two exams (0.5 * Control 1 + 0.5 * Control 2). There will be a second chance to recover that part on the day at Juny we have assigned to. Parts that have not been passed in the partial theory exams may be recovered separately. The minimum grade to pass this part of Theory is > = 5. In the recovery only the failed exams can be recovered, the grade obtained cannot be raised (in the case of passing the Theory part). The maximum grade that can be obtained in recovery is 7.

The part of practical projects (PP) will be evaluated in groups of 2 people. There are two deliveries. The final grade will come from the weighted sum of the two deliveries (0.5 * Delivery 1 + 0.5 * Delivery 2). There will be 2 short and written exams for individual validation of the practical projects (the first one in the class schedule, the second one in the schedule established by the coordination for the second partial exams in June). Practical Projects (PP) = Projects * Validation. To pass the PP the minimum mark will have to be >= 5. There is only one opportunity to pass this part (this part can not be recovered).

The practical exercises (EXER) will be done in groups of 2 people and will consist of workingon very specific programming problems, short and related to those found in laboratory practices. We want each student to study the solution of a set of specific problems in isolation in the most general case of the practice. The value of these exercises is 20% of the final grade and given their nature and purpose are not recoverable.

The final grade of the subject will be the weighted sum of the grades of each of the three activities: 50% of Theory, 30% of practical projects, and 20% Resolution of basic exercises. The result will have to be >= 5.

In case a student does not pass the subject due to not reaching the minimum score in any of the mandatory parts (Theory or Practical projects), even though the weighted average is equal or superior to 5, the final grade of the subject will be 4.5.

In the case that the average does not reach 5, the official grade will be the average mark obtained numerically.

If the student delivers any activity, it is understood that he/she is participating in the subject and will be evaluated. If he/she does not deliver any activity, then it can be considered Non-evaluable.

Granting an honorific matriculation (MH) qualification is a decision of the responsible for the subject. The regulations of the UAB indicate that MH can only be awarded to students who have obtained a final grade of 9.00 or more. It can be granted up to 5% of MH of the total number of students enrolled.

The dates of continuous evaluation and assignment delivery will be published on the communication platform and may be subject to possible changes to adapt to possible incidents; we will always inform about these changes on the moodle platform since it is understood that it is the usual mechanism for exchanging information between professors and students.

For each assessment activity, a place, date and time of revision will be indicated in which the student will be able to review the activity with theprofessor. In this context, claims can be made about the activity grade, which will be evaluated by the professors responsible for the subject. If the student doesnot submit to this review, this activity will not be reviewed later.

SUMMARY

If ((T> = 5) i (PP> = 5) then

NF = 0.50 * T + 0.2 * EXER + 0.30 * PP

If (NF> =5) then PASS

else FAIL

else FAIL

Repeating students: repeating students who have passed laboratory exercices in previous editions of the subject CANNOT request the validation of this part of the subject. All the assessment activities must be carried out under the same conditions as the other students.

Single Assessment Evaluation: This subject DOES NOT HAVE SINGLE ASSESSMENT EVALUATION.

Note about plagiarism:

Without prejudice to other disciplinary measures deemed appropriate, and in accordance with the current academic regulations, irregularities committed by a student who mayleadto a variation of the qualification in an assessable activity will be graded with zero (0). Assessment activities qualified 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 subject, this subject will be suspended directly, without opportunity to recover it in the same course. These irregularities include, among others:

- the total or partial copy of a lab exercice, report, or any other evaluation activity;
- let another student to copy;
- present a group work not done entirely by the members of the group (applied to all members and not only to those who have not worked);
- present as own materials prepared by a third party, even if they aretranslationsor adaptations, and generally works with non-original and exclusive elements of the student;
- unauthorized AI use (i.e. Copilot, ChatGPT o equivalents);
- have communication devices (such as mobilephones, smart watches, pens with camera, etc.) accessible during theoretical-practical assessment tests (individual exams);
- talk with classmates during the individual theoretical-practice tests (exams);

- copy or attempt to copy from other students during the theoretical-practical assessment tests(exams);
- use or attempt to use written material related to the subject during the theoretical-practical evaluation tests (exams), when these have not been explicitly allowed.

Ifyou do not passthe subject dueto the fact that some of the evaluation activities do not reach the minimum grade required, the numerical official grade will be the lowest value between 4.5 and the weighted average of the grades. With the exceptions that the "Non-Appraising" qualification will be awarded to students who do not participate in any of the assessment activities, and that the numerical official grade will be the lowest value between 3.0 and the average Weighted grades in case the student has committed irregularities in an evaluation act (and therefore the subject cannot beapproved by compensation). In future editions of this subject, the student who has committed irregularities in an evaluation act will not be validated any of the assessment activities carried out.

In summary: copy, let copy or plagiarize (or attempt) in any of the assessment activities will lead to a FAIL, not compensable and without validations of parts of the subject in subsequent courses.

Bibliography

Mark Allen Weiss: Data Structures and Algorithm Analysis in C (2nd Edition). Pearson, 1996.

Mark Allen Weiss: Data Structures and Algorithm Analysis in C++. 4 th edition. Pearson, 2014.

Kernighan, Brian; Ritchie, Dennis M., The C Programming Language (2nd edition). Englewood Cliffs, NJ: Prentice Hall. 1988.

K.N. King: C Programming: A Modern Approach. 2nd edition. W.W. Norton & Company. 2008

http://www.cplusplus.com/ : The C++ Resources Network

Software

Windows/Linux

Microsoft Visual Code

Linux or Windows compilers

Queue manager (SLURM)

Remote connection to the laboratory (e.g. MobaTex, Eclipse, Visual Studio, etc.). MobaXTerm for Windows [https://mobaxterm.mobatek.net/] or Visual Studio Code for all platforms [https://code.visualstudio.com/]

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	71	English	second semester	morning-mixed

(PLAB) Practical laboratories	71	English	second semester	morning-mixed
(PLAB) Practical laboratories	72	English	second semester	morning-mixed
(TE) Theory	71	English	second semester	morning-mixed