UAB
Universitat Autònoma de Barcelona

Programming 3

2026/2027

Code: 107892
Credits: 6

Degree programme	Type	Course
Computer Engineering	OB	2

Contact lecturer

Name : Gemma Sanchez Albaladejo

Email : gemma.sanchez@uab.cat

Teaching staff

Roberto Ferrero Pintor

Daniel Soto Alvarez

Group languages

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

Prerequisites

The subject cannot have any official prerequisite by regulation. But students who have not completed and previously passed the subjects of Programació 1 and Programació 2 have a high percentage of suspended. Therefore, it is strongly recommended that the student has satisfactorily completed and passed the previous subjects of Programació 1, Programació 2 as well as Matemàtica Discreta Therefore, you are familiar with the basic and advanced structures of programming, Orientation objects and the concept of graph with the different methods of travel on them.

Objectives

This course is part of the "Algorithms and Information" subject area and should be viewed as the logical continuation of the "Programació 2" course and the practical continuation of the "Matemàtica Discreta" course. The primary objective is to delve deeper into the object-oriented programming concepts introduced in "Programació 2" and expand upon them with additional programming concepts, more complex data structures, and efficient algorithms for traversing them. The course will explore the concept of objects in greater depth by thoroughly examining class templates and inheritance. It will introduce the concept of recursive algorithms, covering both simple and complex examples-such as those related to tree and graph traversals. Furthermore, efficient searching and sorting algorithms will be introduced, and the concepts of time and space complexity will be examined in detail. By the end of the course, students should be able to design and program optimal solutions to complex problems. Accordingly, the learning objectives for the course are as follows:

- To be able to analyze a complex problem, design an optimal solution, implement it, calculate its complexity, and test it.
- To understand and know how to use complex data structures-such as trees and graphs-correctly and efficiently to solve complex algorithmic problems.
- To understand and correctly apply advanced object-oriented programming principles: templates,

abstract classes, and virtual functions. Equip the student with the ability to design algorithms for solving complex problems, covering complex traversal and search algorithms for complex data structures, as well as analyzing their time and space complexity to select the solution best suited to specific needs.
- Introduce the concept of recursion and its application to traversing complex recursive structures, while also enabling the analysis of recursive algorithm complexity.
- Program using a real-world programming language and debug one's own programs.
- Develop programs following style guidelines aimed at ensuring high-quality code. These guidelines include practices that facilitate code comprehension-such as using comments, proper indentation, and meaningful names for variables and functions-as well as the use of exceptions.

Learning outcomes

- CM19 (Create the most appropriate data structures to design and implement algorithms.) Create the most appropriate data structures to design and implement algorithms.
- KM20 (Identify the main data and recursion structures in the definition of algorithms.) Identify the main data and recursion structures in the definition of algorithms.
- SM23 (Analyse the data structures used in algorithm design.) Analyse the data structures used in algorithm design.

Contents

0. Introduction

Course objectives and overview. Review of Object-Oriented Programming and dynamic data structures.

1. Object-Oriented Programming

Advanced object-oriented paradigm. Templates. Inheritance, abstract classes, virtual functions, and polymorphism.

2. Complexity of iterative algorithms

Space and time complexity. Calculating complexity for iterative algorithms.

3. Non-linear data structures: Hashing

Hashing techniques. Hash tables and hash lists. Hash functions.

4. Recursion and sorting algorithms

Introduction to recursive algorithms. Bubble Sort, QuickSort, MergeSort. Recursion. Complexity analysis.

5. Non-linear data structures: Graphs

Representations and traversals. BFS, DFS, problem-solving with graphs.

6. Non-linear data structures: Trees

Tree definition and representation. Tree traversals. Binary Heaps. Red-Black Trees.

7. Python basics

Basic concepts of Python programming.

Learning activities and methodology

Title	Hours	ECTS	Learning outcomes

Independent work	46	1.84	CM19, KM20, SM23
Face-to-face classes	50	2	CM19, KM20, SM23
Prior preparation for classes.	34	1.36	CM19, KM20, SM23
tutorials	1	0.04	CM19, KM20, SM23
Individual Study	13	0.52	CM19, KM20, SM23

The course's teaching methodology is based on the principle that "programming is the only way to learn to program" and, consequently, focuses primarily on practical student work. It also aims to make the most of the in-person time students spend with the instructor. Thus, descriptive theoretical concepts-which students can easily grasp by watching videos or reading articles-will be covered through autonomous (yet guided) study. In contrast, the practical application and expansion of these concepts will take place in class with the instructor's assistance. The course's main objective is for students to be able to solve a given problem efficiently, using complex data structures where necessary. Therefore, the learning process will focus on guiding students through problem-solving tasks based on theoretical concepts they have previously studied independently. The C++ programming language will be the primary tool used, though some concepts regarding Python programming will also be introduced.

The general methodology of the course can be divided into three phases:

- Class preparation: the aim of this phase is for students to learn the concepts to be covered in the next session through various activities provided by the teaching staff, such as watching videos, reading texts, etc.
- In-person class: the goal of this phase is to consolidate the concepts covered and apply them within the context of the course. The teaching staff will ensure that students deepen their understanding of these concepts through exercises-guided to varying degrees-during the session, adding new nuances to the concepts learned independently whenever necessary. This approach makes much better use of the in-person class time, as students are already familiar with the descriptive concepts; this allows the session to focus fully on their application and expansion-areas where students are most likely to need the instructor's assistance. Students must bring a laptop to in-person classes unless the classrooms are already equipped with computers.
- Independent work: to gain proficiency in using complex structures and their associated algorithms, students must complete part of the work on their own, whether through individual exercises or as part of a project.

Project work must be carried out in pairs.

Course management will be handled via the virtual campus platform (https://cv.uab.cat/) and possibly Caronte (http://caronte.uab.cat/).

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

Continuous assessment activities

Title	Weight	Hours	ECTS	Learning outcomes
Second Midterm Exam	35% of the final grade (50% of the theory grade)	2	0.08	CM19, KM20, SM23
Make-up exam	see the description of the assessment method	2	0.08	CM19, KM20, SM23
First Midterm Exam	28% of the final grade (40% of the theory grade)	1.8	0.072	CM19, KM20, SM23

Programming Project	20%	0	0	CM19, KM20, SM23
Delivery of problems	10%	0	0	CM19, KM20, SM23
Third Midterm Exam	7% of the final grade (10% of the theory grade)	0.2	0.008	CM19, KM20, SM23

The course assessment will take into account three types of evaluation activities: Avaluació Individual (individual assessment), Projecte Total de programació (Completed Project), and Avaluació de Problemes (Problem-solving Assessment). Additionally, there is a component for Nota Addicional Problemes Classe (Class Problem-Solving Marks) that can contribute up to 1 point to the final grade, provided the course has already been passed.

The Nota Final (Final Grade) for the course is obtained by combining the assessment of these three activities (plus the additional one) as follows:

Nota Final = (0.7 * Avaluació Individual) + (0.2 * Projecte Total) + (0.1 * Avaluació Problemes) + Nota Addicional Problemes Classe (màxim 1 punt)

Final Grade = (0.7 * Individual Assesment) + (0.2 * Completed Project) + (0.1 * Problem-solving Assessment) + Class Problem-Solving Marks (1 maximum point)

The various components of the course grade are detailed below:

- Avaluació Individual (Individual Assessment): This section covers the individual assessments conducted throughout the course. The assessment will consist of three partial exams and a final make-up exam. The first partial exam will take place during the teaching term, preferably during a time slot that allows for a common schedule across all groups taking the course. The remaining partial exams and the make-up exam will be scheduled in accordance with the calendar and timetable established by the degree program coordination.

The final retake exam will allow students to retake any failed partial exams and, optionally, improve the grades of partial exams they have already passed. Students who have not passed one or more partial exams will only be required to take the exam for the content corresponding to the failed sections. Additionally, students wishing to improve the grade of a previously passed partial exam may also voluntarily take the retake exam. In all cases, the final grade for each partial exam will be the highest of the scores obtained between the regular exam session and the retake exam.

The Individual Assessment grade will be calculated according to the following formula:

Avaluació Individual = (0,4×Parcial 1) + (0,5×Parcial 2) + (0,1×Parcial 3)

Individual Assessment = (0,4×Partial 1) + (0,5×Partial 2) + (0,1×Partial 3)

To pass the course, it will be necessary to:

- Obtain a minimum grade of 4.0 on each of the three partial exams.
- Obtain a minimum Avaluació Individual (Individual Assessment) score of 5.0.

- Projecte Total (Project Total): This section covers the work carried out for the programming project, which is to be completed in groups of up to two people. The project grade will be based on the assessment of the various deliverables and on an individual progress test designed to verify each student's participation and level of knowledge regarding the project developed.

The final project grade will be calculated using the following formula:

Projecte Total = (0.2 x Avaluació Seguiment Projecte) + (0.3 x Entrega Parcial 1) + (0.5 x Entrega Final)

Project Total = (0.2 x Project Monitoring and Evaluation) + (0.3 x Partial Submission 1) + (0.5 x Final Submission)

where:

- Partial Submission 1 (30%): mid-course project interim submission.
- Final Submission (50%): final version of the project.
- Project Monitoring and Evaluation (20%): Individual test on the completed project. The continuous assessment will consist of an individual test taken alongside the course's second and third midterms. If a student does not pass, a retake opportunity will be available on the day of the final make-up exam.

To successfully complete the project, it will be necessary to:

- Obtain a minimum grade of 4.0 in the Project Monitoring and Evaluation.
- Obtain a minimum grade of 5,0 in the Entrega Final.
- Obtain a minimum grade of 5,0 in the grade of Project Total.

The grade for the final submission may be retaken only if:

- The Project Total grade is 3.0 or higher, and
- The Individual Assessment grade is 5.0 or higher. This retake of the final submission will take place on the dates scheduled for the course retake.

- Avaluació Problemes (Problem-solving Assessment):This section includes the exercises identified as AVALUABLES (ASSESSABLE) that will be assigned throughout the course and must be completed individually and independently.

A minimum grade for this activity is not required to pass the course. However, this component accounts for 1 point of the final grade, so failing to submit the required work could prevent you from achieving the minimum grade needed to pass the course.

Assignments submitted after the deadline, or those that received a failing grade, may be resubmitted up to the date of the course's final exam. In such cases, the maximum grade obtained will be subject to a 20% penalty.

The contribution of each problem to the grade for this section will be calculated in proportion to the topic's weight within the course and the number of assessable problems associated with each topic.

- Nota Addicional Problemes Classe (Class Problem-Solving Marks) (maximum 1 point): During class sessions, students will complete exercises that can be submitted via the digital platform. These exercises will allow them to earn up to one additional point towards the final course grade.

Two submissions will be required for each exercise:

- First Submission: It must be done at the end of the class session. The exercise does not need to be fully functional, but it must contain code developed during the session. This submission is mandatory in order to be eligible for the final submission.
- Final Submission: It may be completed until 11:59 PM on the day following the session. Only students who submitted the first assignment within the established deadline may do so.

These submissions cannot be made up. The contribution of each exercise to the additional score will be determined based on the topic's weight within the course and the number of submission-based exercises associated with that topic.

Non-assessable: Students will be considered non-assessable (NA) if they do not submit at least 50% of the exercises and do not take any of the following assessment tests: midterm 1, midterm 2, midterm 3, final retake exam, or final practical assignment submission.

Failures: If the calculated final grade is 5 or higher but the minimum requirement for any of the assessment activities is not met, the final result will be a fail, and a grade of 4.5 will be recorded in the academic record of

students in this situation.

Convalidation: For students repeating the year, the project grade from the previous year (2025-26 academic year) will be recognized if the following conditions are met:

a)The Final Grade for the previous year's Project Total is 7 or higher.

b)The grade for the previous year's Individual Assesment is 3 or higher.

MH: As many Distinctions as possible will be awarded within university regulations, starting with the highest grades and provided the minimum grade is a 9.

Reviews: For each assessment activity, a location, date, and time will be specified for students to review the activity with the instructor. Some of these reviews may also be conducted via the virtual campus using the pilot exam review environment. In this context, students may submit appeals regarding their activity grade, which will be evaluated by the course instructors. Students who do not attend the scheduled review session will not have the activity reviewed at a later time.

Protocol for requesting the rescheduling of assessment activities:

If a student is unable to attend a midterm exam or the individual project progress assessment for a reason duly justified and recognized by the degree program, no alternative exam will be scheduled prior to the retake exam. In such cases, the student will take the corresponding retake exam directly. Only if the student fails to demonstrate the achievement of learning outcomes through this exam will the teaching team consider conducting an additional, specific assessment activity.

Justified issues preventing the submission of problem sets or the project will be analyzed individually by the teaching team, which will determine appropriate compensatory measures based on the specific circumstances of each case. In the event of general technical issues affecting the submission platform that prevent or significantly hinder the submission of assignments, class representatives must notify the entire teaching team of the issue as soon as possible. Once the issue has been verified, an extraordinary extension to the submission deadline will be granted, proportional to the duration of the service disruption.

Important note regarding copying and plagiarism:

Without prejudice to any other disciplinary measures deemed appropriate, and in accordance with current academic regulations, irregularities committed by students that could lead to a change in the grade will result in a score of zero (0). Assessment activities graded in this manner and through this procedure cannot be retaken. If passing any of these assessment activities is required to pass the course, the student will automatically fail the course, with no opportunity to retake it during the same academic year. Such irregularities include, among others:

- Copying all or part of a practical assignment, report, or any other assessment activity;
- Allowing others to copy one's work;
- Submitting group work that was not produced entirely by the group members;
- Presenting materials created by a third party as one's own-even if they are translations or adaptations-and, in general, submitting work containing elements that are not original to and exclusively the work of the student;
- Having communication devices (such as mobile phones, smartwatches, etc.) accessible during individual theoretical-practical assessments (exams).

In these cases, the numerical grade for the course will be the lower of the two values: 3.0 or the weighted average of the grades (and, consequently, passing by compensation will not be possible). Tools for detecting code plagiarism will be used when evaluating problem sets and practical assignments.

Note on the planning of assessment activities. Dates for continuous assessment and assignment submissions will be published at the start of the course and may be subject to schedule changes to accommodate potential issues. Notifications regarding such changes will always be provided via Caronte and/or the Virtual Campus, as these are the standard platforms for information exchange between lecturers and students.

Single assessment. This course does not provide for a single assessment system.

Use of AI. Restricted use: For this course, the use of Artificial Intelligence (AI) technologies is permitted exclusively for support tasks, such as bibliographic or information searches, text proofreading or translation, generating short class initialization code, or simple processes involving the traversal of basic structures like vectors. Students must clearly identify which parts were generated using this technology, specify the tools employed, and include a critical reflection on how these tools influenced the process and the final outcome of the activity. Failure to be transparent about the use of AI in this assessable activity will be considered a lack of academic honesty and may result in a partial or total grade penalty, or more severe sanctions in serious cases.

Bibliography

- http://www.cplusplus.com/ : The C++ Resources Network
- https://es.wikibooks.org/wiki/Programaci%C3%B3n_en_C%2B%2B: Programación en C++ - Wikilibros
- https://www.geeksforgeeks.org/c-plus-plus/?ref=ghm
- Mark Allen Weiss. Data Structures and Data Analysis in C++. Pearson. 2014.
- B. Eckel. Thinking in C++, Volume 1: Introduction to Standard C++, Prentice-Hall, 1999
- B. Eckel. Thinking in C++, Volume 2: Standard Libraries and Advanced Topics, Prentice-Hall, 1999
- F. Xhafa, P. Vázquez, J. Marco, X. Molinero, A. Martín: Programación en C++ para ingenieros. Thomson, 2006
- Scott Meyers. Effective Modern C++: Specifics Ways to Improve Your Use of C++11 and 14. O'Reilly Media, Incorporated, 2014.
- Robert C. Martin. Código limpio : manual de estilo para el desarrollo ágil de software. Anaya Multimedia, 2012
- Thinking in PYTHON Bruce Eckel (se puede descargar de http://www.bruceeckel.com).
- Learning PYTHON 2nd Edition. Mark Lutz and David Ascher, Safari Tech Books Online.
- Manuals de Python (de la pagina web oficial).
- https://www.geeksforgeeks.org/python-programming-language/?ref=ghm
- Llibres electronics interactius de python:
- http://interactivepython.org/runestone/static/thinkcspy/toc.html#t-o-c
- http://interactivepython.org/runestone/static/pythonds/index.html
- http://www.pythontutor.com/

Software

- Microsoft Visual Studio
- Spyder Anaconda

Course groups and languages

The information provided is provisional until November 30. After this date, you will be able to consult the language of each group through this link. To access the information, you will need to enter the course CODE

Type of teaching	Group	Language	Semester	Shift
(PAUL) Classroom practices	411	Catalan/Spanish	first semester	morning-mixed
(PAUL) Classroom practices	412	Catalan/Spanish	first semester	morning-mixed
(PAUL) Classroom practices	431	Catalan/Spanish	first semester	morning-mixed
(PAUL)	432	Catalan/Spanish	first	morning-mixed

Classroom practices			semester	
(PAUL) Classroom practices	451	Catalan/Spanish	first semester	morning-mixed
(PAUL) Classroom practices	452	Catalan/Spanish	first semester	morning-mixed
(PAUL) Classroom practices	453	Catalan/Spanish	first semester	morning-mixed