UAB
Universitat Autònoma de Barcelona

Techniques of Algorithm Design

Code: 104393
ECTS Credits: 6

2025/2026

Degree	Type	Year
Computational Mathematics and Data Analytics	OB	2

Contact

Name: Enrique Ruiz Amakrache

Email: enrique.ruiz@uab.cat

Teaching groups languages

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

Prerequisites

Although it is not compulsory, it is highly advisable to have studied the following subjects:

- Introduction to Programming
- Algorithm and Combinatory in Graphs. Heuristic Methods
- Object-oriented programming

It will be considered that the student has already acquired the knowledge imparted in these subjects.

Objectives and Contextualisation

Based on the basis that students have basic knowledge about programming and data structures, it is intended that the student is able to analyze, design and implement algorithms based on the design techniques of existing algorithms. To fulfill this objective, the student will acquire the knowledge about:

- Formal problem specification. How to move from a description of a problem to a valid specification for the development of an algorithm that solves it.
- Formal tests to validate programs. Design based on contracts. Preconditions, postconditions and invariant.
- Complexity analysis.
- Paradigms for the design of algorithms. Greedy. recursion, backtracking, branch & bound, dynamic programming, probabilistic algorithms, etc.

The development of an algorithm begins by formalizing the statement of a problem. From this statement an algorithm is designed to solve the problem, but this is not enough, we must consider how long it will take the algorithm to give us the solution. So we are interested in creating algorithms as fast as possible. In this way we can create programs that solve problems as large as possible in acceptable times. This speed is achieved by designing algorithms that minimize the number of operations to be performed to solve a problem and developing an efficient implementation of the algorithm's operations. This will mean that in this subject the student will acquire algorithmic knowledge and efficient implementation of algorithms.

Learning Outcomes

1. CM28 (Competence) Design effective algorithmic solutions to computational problems in accordance with the established requirements.
2. CM29 (Competence) Assess the computational complexity of the algorithmic solutions to develop and implement the one that guarantees the best performance.
3. KM23 (Knowledge) Identify the suitable programming strategies to solve a given problem.
4. SM24 (Skill) Implement recursive solutions to programming problems.
5. SM26 (Skill) Apply the fundamental principles and basic techniques of parallel, concurrent and distributed programming.

Content

The syllabus of the subject will be:

- Formal specification of algorithms
- Algorithmic Complexity
- Greedy Algorithms
- Recursion
- Divide and conquer
- Backtracking
- Branch & bound
- Dynamic programming
- Probabilistic Algorithms

Activities and Methodology

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Classes (theory, problems and practices)	50	2	
Type: Supervised			
Monitoring in the assimilation of theoretical concepts	4	0.16	
Reinforcement and monitoring in the resolution of projects	4	0.16	
Type: Autonomous			
autonomous work	12	0.48	
Intermediate exam preparation	12	0.48	
Pre-class preparation	48	1.92	
Preparation of practical reports	8	0.32	

Taking into account that the final objective of the subject is that students are able to analyze and design algorithms efficiently according to a given problem, the work of the students is the central axis of their learning, accompanied and guided by the teaching staff. For this reason, the face-to-face classes will be highly practical and will focus on the students consolidating the knowledge that is the learning objective of this subject.
The general methodology of the subject can be divided into three phases:

Preparation of the class: the objective of this phase is that the students can learn the concepts that will be worked on in the following session through various activities offered by the teaching staff, such as viewing videos, reading texts, etc.

Class: the objective of this phase is to consolidate the concepts seen and put them into value within the context of the subject. The faculty will ensure that the students deepen these concepts through exercises (more or less) guided during the session. Thus, the class will begin with a summary explanation of the concepts that the student will have studied in the preparation of the class. This summary will give an overview about the concepts under study and will allow students to solve any doubts they may have. Then the problem or practical work to be done during the second part of the class will be explained. These problems or practices will be carried out with a computer, since in many cases they will involve implementing algorithms.

Autonomous work: in order for students to be fluent in the programming of the algorithms they have seen, they will have to do some of the work on their own, whether they are individual exercises or within a project.

To make learning based on problems and practices more dynamic, intensive use of automatic correctors will be made, whenever possible. Thus, for each problem or practice that the student has to deliver, they will be provided withan automatic corrector that will allow them to immediately test the program they are developing. Thus, the student will know if he has solved the problem or there is still some error that has to be solved. In addition, to put emphasis on the optimal implementation of algorithms, the student will be able to compare the execution times of his solution with that provided by the teacher. Thus, we want to gamify the practical work, marking as objective and rewarding overcome the teacher's solution.
For the realization of the problems and practices, the C ++ programming language will be used at the basic level. This language is selected because it is the high level language that obtains the maximum performance of the current processors, therefore, it is especially appropriate for the optimal implementation of algorithms.

Adaptation of the methodology to non-contact teaching

Non-contact teaching will give more importance to the autonomous work of the student and implies a methodological change where the inverted class will be applied. Considering this concept, we will plan the weekly teaching as videos / pdfs with the theoretical subject of the week. From 4 hours of face-to-face teaching there will be 2 hours of online class. In these classes, problems will be solved and progress will be made in the development of the practice. At the end of the week there will be a delivery of problems or practices so that the teacher can check the follow-up of the subject by the students.

Class preparation: Each week will begin with the study or review of the material that the teacher will publish on the virtual campus. Normally this material will be a presentation made with powerpoint and a video where the pages of the presentation will be commented as if it were a typical master class of theory. This material will explain theoretical concepts, give examples of problem solving and propose problems for online and delivery classes. The student must study this material before the online classes.

Online class: The teacher will make a short introduction to the problem / practice that will be discussed during the class. From this moment on, a dialogue must be established where students will express their doubts and propose how to solve the problem / practice. The teacher will guide the students to solve the problem based on the students' proposals. Where possible, student / teacher solution proposals will be implemented in class. Class participation will be scored and will serve to improve the grade of the subject.

Autonomous work: Autonomous work will be the same as in the face-to-face methodology.

Material needed for online teaching

The Microsoft Teams application, which must be installed on the student's computer, will be used for both online and consultation classes. This computer should have a camera and microphone to facilitate interaction in classes and consultations. Visual Studio C ++ and the Microsoft Windows operating system will be used for programming.

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
Delivery of exercises	See description assessment	1	0.04	CM28, CM29, KM23, SM24, SM26
Final Exam (Recovery)	See description assessment	4	0.16	
First Partial Theoretical-Practical Exam	See description assessment	2	0.08	CM29
Group project evaluation	See description assessment	2	0.08	CM28, KM23, SM24, SM26
Individual evaluation of the project	See description assessment	1	0.04	CM29
Second Partial Theoretical-Practical Exam	See description assessment	2	0.08	CM29

Criteria and evaluation indicators:

- Understanding of the theoretical concepts of the subject.
- Correct and efficient implementation of algorithms.

Activities and evaluation instruments:

Note Theory: corresponds to two partial exams of theory on the subject. Partial exams have been approved to eliminate subject in the recovery exam.
Note of problems: Corresponds to the grade obtained by a series of problems that the student will have to deliver. There is no recovery for the delivery of problems.
Note Practices: In this section there is a group note and an individual note:
Group note: Corresponds to the grade obtained to the deliveries by groups of the practice. Deliveries of suspended practices can be reoccurred in subsequent deliveries.
Individual note: It is the score obtained in the practice exam. There will be recovery of the practice exam.

The final grade of the subject is obtained by combining the evaluation of these three activities as follows:

IF Note Theory> = 4 and Practice Note> = 4 THEN

Final Note = 0.4 * Note Theory + 0.2 * Note problems + 0.4 * Practice Note

ELSE Final Note = MIN (Theory Note, Practice Note)

IF First partial exam note> = 4 and Second partial exam note> = 4 THEN

Note Theory = 0.5 * First partial exam grade + 0.5 * Second partial exam note

ELSE Note Theory = MIN (Note part-time exam, Second part exam note)

Note problems = Weighted average of the problems delivered.

```
IF Individual Note> = 4 and Note Group> = 4 THEN
    Practices Note = 0.2 * Individual Note + 0.8 * Group Note
ELSE Note Practices = MIN (Individual Note, Group Note)
```

Individual Note = Exam of practices.

Note Group = Weighted average of the deliveries of practicesif all are approved, but the minimum of the notes of the deliveries.

Validation of practices: practices of previous years are not validated.

Recovery of practices: In the case of having suspended a group delivery, it may be recovered in the following installments of the practice. The note will be 0.8 * (max (recovery note, delivery note suspended) -Note delivery suspended) + delivery note suspended. In the case of suspending the internship exam, the student must submit to a practice recovery exam on the same day of the final recovery exam.

Conditions to pass the subject:

To pass the subject the student must have passed:

Final note>=5
Note theory>=4
Practical notes>=4
Notes delivery of practices>=4
Exams>=4

Conditions for non-evaluable:

Not having any part of the subject suspended.

Conditions for suspense:

- Do not achieve an average grade greater than or equal to 5.
- Suspend any of the evaluation activities of the subject, even if the average exceeds 5. In this case, the grade will be the minimum grade obtained from any of the parts (exams or practices).

Conditions for honors registration:

- Honors can be obtained with an average score greater than or equal to 9.0.
- Because there are a limited number of honor plates that can be given per group, they will be awarded in order of grade from highest to lowest.

Practices, works or exams copied:

Without prejudice to other disciplinary measures deemed appropriate, and in accordance with current academic regulations, irregularities committed by a student that may lead to a variation of the grade will be scored with a zero (0). 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. These irregularities include, among others:

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

- let copy;

- present a group work not done entirely by the members of the group;

- present as own materials prepared 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, smart watches, etc.) accessible during the theoretical evaluation-individual practices (exams).

In case the student has committed irregularities in an evaluation act, the numerical note of the file will be the lower value between 3.0 and the corresponding grade according to the method of evaluation of the subject (and therefore it will not be possible to pass the compensation).

In summary: copy, let copy or plagiarize in any of the evaluation activities is equivalent to a FAIL with a grade lower than 3.0.

Publication of notes, exam dates, etc:

The dates for continuous evaluation and submission of papers will be published on the virtual campus and may be subject to changes in programming for reasons of adaptation to possible

Unique assessment:

It is the same as the normal assessment, but all papers will be presented on the day of the subject's final exam. Therefore, it consists of:

Examination of the first partial theory.
Examination of the second partial theory.
Practice exam.
Issue deliveries (non-recoverable).
Practice deliveries (non-recoveable).

Bibliography

Fundamentos de Algorítmia, G Brassard P. Bratley. Prentice Hall.

Técnicas de Diseño de Algoritmos, Rosa Guerequeta y Antonio Vallecillo.

Técnicas de diseño de algoritmos, F. Perales, M. Mascaró. Universitat de les Illes Balears

Thinking in C++ 2nd Edition by Bruce Eckel Volume 1 y Volume 2.

(http://www.mindview.net/Books/TICPP/ThinkingInCPP2e.html)

El lenguaje de programación C++, Bjarne Stroustrup

http://en.cppreference.com/w/

http://www.cplusplus.com/

Software

The practice is done with a Windows 10 computer. The Visual Studio 2022 compiler is used to compile C ++ programs. In addition, a project is used for practice, which is a graphical application and automatic correctors that are provided from the virtual campus.

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
(PLAB) Practical laboratories	1	Catalan/Spanish	first semester	morning-mixed
(TE) Theory	1	Catalan/Spanish	first semester	morning-mixed