Universitat Autònoma de Barcelona

Fundamentals of Programming II

2021/2022

Code: 106485
ECTS Credits: 6
Type: FB Year: 1 Semester: 2

Degree

1488 - Artificial Intelligence

The contents of this guide are provisional and may be subject to minor changes. The final version of the guide will be available at the beginning of the semester.

The proposed teaching and assessment methodology that appear in the guide may be subject to changes as a result of the restrictions to face-to-face class attendance imposed by the health authorities

Contact

Name: Elisa Heymann

Email: Elisa.Heymann@uab.cat

Use of languages

Principal working language: English

Some groups entirely in English: Yes

Some groups entirely in Catalan: No

Some groups entirely in Spanish: No

Prerequisites

The course doesn’t have any official prerequisites. Nevertheless, it builds on top of Fundamentals of Programming I. Therefore, the student should be familiar with the basics of programming and Python.

Objectives and Contextualization

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

- Understanding different data structures such as stacks, queues, and lists; and more complex structures such as hast 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

- To design, implement, analyse and validate efficient and robust algorithmic solutions to computational problems derived from the design of intelligent systems.
- To know and efficiently use the techniques and tools of representation, manipulation, analysis and management of large-scale data
- To develop critical thinking to analyse in a grounded and reasoned way alternatives and proposals, both own and others.
- To work cooperatively to achieve common objectives, assuming own responsibility and respecting the role of the different team members.
- To analyse and solve problems effectively, generating innovative and creative proposals to achieve the objectives.

Learning Outcomes

- To select and use the appropriate algorithmic and data representation structures to solve a problem.
- To apply the principles of object-oriented programming in program development.
- To develop programs with a good programming style and well documented.
- To apply debugging, testing and correction strategies to verify the correct operation of an algorithmic solution according to the requirements of the problem to be solved.
- To implement recursive solutions to programming problems.
- To Evaluate and analyze the computational complexity of algorithmic solutions in order to develop and implement the one that guarantees the best performance.
- To understand common data structures and the algorithms that use and manipulate them.
- To know the limitations and advantages of different data representations and the ability to choose the most efficient structures that facilitate the subsequent analysis and exploitation of the data.

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.
- Tress.
- Analysis of the complexity of algorithms.

Methodology

We will use the Caronte platform (http://caronte.uab.cat/) to share materials, deliver projects, and other administrative activities. To enroll in Caronte, you must register (with name, NIU, and photo) and enroll de subject (the subject code will be 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. More specifically, the different types of teaching activities are the following:

- Theoretical classes. Aimed at explaining the different topics of the course, using many examples.

- Problem-solving classes: Joint resolution of problems. The problem set will be available in Caronte, and will be self-assessed.

- 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.

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.

Activities

Title	Hours	ECTS	Learning outcomes
Type: Directed			
Theoretical lessons	25	1	
Problem solving seminaries	25	1	
Type: Supervised			
Type: Autonomous			
Autonomous work	94	3.76	

Assessment

The evaluation will be continuous.

- Exams: There will be a midterm and a final exam.

- Recovery exam: Students who do not achieve a threshold grade (4) in the theoretical exams, will need to talk a recovery exam on the failed exam.

- Projects (Proj): There will be 2 projects. Each one of the projects must be passed separately. In case of failing any project, there will be the option to recover it. The grade of the recovery will be a maximum of 5.

GRADE INDICATORS:

The final mark is calculated as follows:

- Theory grade = 0'5*Midterm +0'5*Final
- Project grade = 0’5*Proj1 + 0’5*Proj2
- COURSE FINAL GRADE = 0'5 * Theory Mark + 0'5 * Project Mark
-

EVALUATION CRITERIA

- For considering the *Theory grade*, it is necessary to obtain a minimum mark of 4 in each partial to make the average.
- For considering the *Project grade*, it is necessary to obtain a minimum grade 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 FINAL SCORE 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.
- Project deliveries: 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.

Assessment activities

Title	Weighting	Hours	ECTS	Learning outcomes
Midterm exam	25%	2	0.08	
Final exam	25%	2	0.08	
Projects	50%	2	0.08	

Bibliography

- Mark Allen Weiss: Data Structures and Data Analysis in C++. 4th edition. Pearson, 2014.
- Kernighan, Brian; Ritchie, Dennis M., The C Programming Language (2nd edition). Englewood Cliffs, NJ: Prentice Hall. 1988.
- http://www.cplusplus.com/ : The C++ Resources Network

Software

Microsoft Visual C++.
Visual Studio. Available for UAB students at https://msdnaa.uab.cat/