Universitat Autònoma de Barcelona

2020/2021

Basic Computing

Code: 103819
ECTS Credits: 9

Degree	Type	Year	Semester
2501233 Aeronautical Management	FB	1	A

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: Josep Velasco González
Email: Josep.Velasco@uab.cat

Use of Languages

Principal working language: catalan (cat)
Some groups entirely in English: No
Some groups entirely in Catalan: Yes
Some groups entirely in Spanish: No

Teachers

Lluís Ribas Xirgo

Josep Sole Pareta

Prerequisites

There are no prerequisites for this subject, although it is advisable to have some experience in the use of computers.

Keep in mind that this is a subject of 9 credits, equivalent to about 270 hours of work on average, including classes and lab sessions. Therefore, on average, during the first semester a student must dedicate about 10 hours per week of work and, during the second, about 5 hours per week.

Objectives and Contextualisation

This subject has a general and introductory character to computer science. From the perspective of computers as machines capable of processing information automatically, the aim is to achieve the following goals:

- Familiarize students with the computer as a tool for computer systems.
- Provide an overview of computing by explaining the basic concepts: hardware, software, operating system, computer structure, algorithm, program and programming languages, compiler, interpreter, etc.
- To equip students with the ability to design algorithms for problem solving, progressively and systematically introducing a rigorous programming methodology, which is based, fundamentally, on the technique of top-down algorithm design.
- Introduce a real programming language to students. They are intended to perceive the difference of natural language and the strict rules of a real programming language, both in its lexical aspects (valid words of language) and syntactic (rules to combine them) as semantic (meaning of the phrases).
- Make the student used to develop programs following style standards that allow to achieve quality programs. This regulation includes, for example, rules that make them more intelligible such as the use of comments, the bleeding to the right of certain parts of the code, and the use of appropriate names for the data.

- Make the software lifecycle understandable: problem analysis (understand what to solve), design (propose an algorithmic solution), implementation (coding in a programming language of the chosen solution) and verification (systematic testing to ensure that the implemented solution is correct).
- Show the rudiments of databases and their practical involvement.
- Understand how operating systems work.
- Provide an overview of the computer's functional units and their interconnection.
- Provide a knowledge of the operation of the computer at low-level.
- Show the different levels of hardware and software necessary for the operation of computers and their interconnection.

Competences

- Apply specific software for solving problems in the aeronautical sector.
- Communication.
- Develop software of low or medium complexity.
- Personal work habits.
- Thinking skills.
- Use knowledge of the fundamental principles of mathematics, economics, information technologies and psychology of organisations and work to understand, develop and evaluate the management processes of the different systems in the aeronautical sector.
- Work in teams.

Learning Outcomes

1. Communicate knowledge and findings efficiently, both orally and in writing, both in professional situations and with a non-expert audience.
2. Develop critical thought and reasoning.
3. Develop systemic thinking.
4. Develop the ability to analyse, synthesise and plan ahead.
5. Identify and use the basic principles of algorithmics.
6. Identify the basic components of a computer.
7. Programme and integrate routines in a high-level language.
8. Understand the principles behind the functional structure of an operating system.
9. Use the software tools studied on the course.
10. Work cooperatively.
11. Work independently.

Content

1. Algorithms and programming
1.1. Algorithmic. Programming languages.
1.2. Imperative model: sequence, selection and iteration.
1.3. Algorithmic schemes: exploration and search.
1.4. Modular design and refinement
1.5. Functions
1.6. Vectors and tuples
1.7. Basic methods of sorting and processing
1.8. Queues and stacks
1.9. Files

2. Data structures
2.1. External data structures: devices, organization, access.
2.2. Databases

3. Representation of information
3.1. Binary encoding of information

3.2. Numbering systems
3.3. Binary arithmetic

4. The computer
4.1. Computer architecture
4.2. Operating systems
4.3. Office automation software

Methodology

Teaching will be offered on campus or in an on-campus and remote hybrid format depending on the number of students per group and the size of the rooms at 50% capacity.

Teaching is based on the following activities:

- Theory classes, in which an exhibition of the theoretical part of each subject of the program will be made.

- Problem seminars. Most topics are accompanied by a relationship of exercises that students must solve at home. The seminars will review the most critical aspects of understanding and solving problems. Those the teacher considers most interesting or the students find more difficult will be developed in detail.

- Laboratory practices. There will be a subject project that must be developed as a team. A guideline will be available to establish the work that should be done prior to laboratory sessions. The project will be evaluated by team, but also individually, so all members of each team must actively participate in the development of the project.

All activity support material will be accessible so to take full advantage of classes and to support individual and group activities performed.

The "learning path", with the description of the activities and the resources necessary to carry them out are available on the UAB Virtual Campus (CV), which will be the usual channel for exchanging information between teachers and students outside the classroom.

TRANSVERSAL COMPETENCES

CT1.1 Develop critical thinking and reasoning. The solution to case studies that students work with is not unique. Combining the tools provided and the logic must be achieved an optimal or, at least, reasonable solution. This competence is therefore evaluated by assessing the quality of the solutions provided to the case studies in which they work.

CT1.2 Develop analytical, synthesis and prospective capacity. This competence is inherent to the resolution of the exercises anddevelopment of the projects of the subject. It is evaluated by assessing the understanding of the scope of the problems that arise and the validity of the solutions that are provided.

CT1.4. Developing systemic thinking: It is evaluated with the follow-up of subject projects.

CT2.1 Work autonomously. To achieve an understanding of the theoretical contents of the subject you must work individually. Self-employment is evaluated through written evidence and by resolving proposed exercises.

CT3.1. Work cooperatively: The practical projects of the subject are developed in work teams. The division of tasks and decision-making systems that each team has adopted is evaluated.

CT4.1. Efficiently communicate, orally and in writing, knowledge, results and skills, both in professional and non-expert audiences. It is evaluated through project follow-up presentations and through written tests.

Activities

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Laboratory practices sessions, with presentation of projects and evaluation of results	12	0.48	3, 7, 10, 9
Master classes, with demonstrations and problem-solving	37.5	1.5	8, 6, 7
Problem seminars with problem solutions, discussion of problems and resolution of doubts	19.5	0.78	3, 4, 2, 7, 10, 9
Resolution of practical cases presented by teams	65	2.6	7, 10, 9
Type: Supervised			
Follow-up and reinforcement in the resolution of parts of the cases proposed in the practices	6	0.24	7, 10, 9
Tutorials with resolution of additional problems	2	0.08	7
Type: Autonomous			
Individual study of the theoretical part	33	1.32	8, 2, 6, 11
Problem solving individually or as a group	39	1.56	4, 5, 7, 11, 9

Assessment

a) Scheduled evaluation processes and activities

The evaluation of the subject is continuous through written tests, delivery of written works and practical work in mandatory attendance sessions and evaluated in the laboratory. A minimum compliance is established from which the student is able to pass the course. If these minimums are not reached, the subject is failed.

The mark for the first semester, $s1$, is obtained as follows:

$$s1 = x1 \cdot 40\% + x2 \cdot 40\% + p \cdot 20\%, \text{ whenever } x1 \geq 5 \text{ and } x2 \geq 5$$

Where $x1$ and $x2$ are the marks of partial exams and p the mark obtained by delivering solutions to the proposed exercises. Therefore, it is necessary to score a minimum of 5/10 in each of the partial exams to pass the subject. If the mark of any partial exam is under 5 out of 10, the mark for the first semester, $s1$, is the lowest of the following values: the corresponding weighted average or 4.5 out of 10.

In the second semester the mark depends on the practical work. Attendance at laboratory sessions is mandatory and necessary to pass the practical work and, by extension, the subject. Practical work is carried out by teams, but the evaluation is individual. The mark for the second semester, $s2$, has a weight of 30% on the final score.

The final grade of the course is obtained as follows:

$$n = s1 \cdot 70\% + s2 \cdot 30\%, \text{ whenever } s1 \geq 5 \text{ and } s2 \geq 5$$

While computing n, if a mark is under 5 out of 10, the result is the lowest of the following values: the corresponding weighted average or 4.5 out of 10.

In short, in the first semester, partial exams must be passed separately, and, in the second semester, the practical work performed by teams, but with individual evaluation.

b) Scheduling of evaluation activities

The dates for the continuous evaluation tests of theory, problems or exercises and practical work will be published on the Virtual Campus (CV) and may be subject to possible programming changes for reasons of adaptation to possible incidents: it will always be reported previously through the CV as it is understood to be the usual mechanism for exchanging information between teachers and students outside the classroom.

c) Re-evaluation process

In the first semester, there is a second-chance exam for each partial exam. These two second-chance exams take place the same day (one after the other). On the other hand, undelivered jobs will receive a mark of 0 and will not have a second evaluation.

In the second semester, if laboratory sessions have been attended, but a minimum mark of 5 out of 10 is not accomplished in the lab practical work, you are eligible for a second-chance test. This test, however, is in no way a substitute for the practical lab sessions. That is, you are not eligible for the second-chance test if you have not attended regularly lab sessions.

d) Qualification review procedure

For partial exams and for the final second-chance exam, a specific day and time will be set for review of corrections. All other evaluated activities can be reviewed at the teacher's tutoring hours.

If, as a result of a review, a change on a mark is agreed, that mark cannot be modified again in a later review.

e) Special grades

If no delivery is made, no lab session is attended and no exam is taken, the corresponding grade is a "not gradable". In any other case, "not presented" computes as a 0 for the calculation of the weighted average. In other words, participation in an evaluated activity implies that "not presented" activities are considered as zeros. For example, an absence in a lab session involves a zero for that activity.

Distinction grades shall be awarded to those who obtain a final grade greater than or equal to 9.5 out of 10, up to a 5% of those enrolled in the descending order of the final grade. At the discretion of the teachers' staff, other cases may also be awarded, provided that it does not exceed 5% of the total enrolled students and that the final grade is 9.0 out of 10 or more.

f) Irregularities committed by the student, copying and plagiarism

Copies refer to evidence that the work or exam has been carried out in part or in full without the intellectual contribution of the author. This definition also includes proven attempts to copy exams and reports or work deliveries, as well as violations of the standards that guarantee intellectual authorship. Plagiarism refers to the works and texts of other authors that are passed as their own. It is an intellectual property crime. To avoid incurring in plagiarism, the sources used when writing a report should be cited.

In accordance with UAB regulations, both copies and plagiarism or any attempt to alter the outcome of the evaluation, own or alien -allowing to copy, for example, imply a final score of the corresponding part of 0 and a fail of the subject in the case of partial exams, final tests or practical work. Evaluation activities qualified in this way and by this procedure will not be recoverable, and therefore the subject is failed directly without any kind of second chance in the same academic year.

h) Evaluation of repeat students

No differentiated treatment has been established for students repeating the subject.

Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes

First partial exam	35	2	0.08	1, 5, 7, 11
Practical work	30	7	0.28	1, 3, 4, 2, 7, 10, 9
Second partial exam	35	2	0.08	8, 1, 4, 2, 6, 5, 7, 11

Bibliography

[1] A. Prieto, A. Lloris, J.C. Torres. (2006). Introducción a la informática. 4rta. Edició. McGraw-Hill.
[2] J.A. Pérez López, Ll. Ribas Xirgo. (2006). Introducció al desenvolupament de programari. Chapters 1, 2 i 3, from 3.1 to3.6. UOC OpenCourseWare.
[http://ocw.uoc.edu/informatica-tecnologia-i-multimedia/introduccio-al-desenvolupament-de-programari/materials/
[3] F. Virgós, J. Segura. (2008). Fundamentos de Informática. McGraw-Hill. [www.mhe.es/virgos]