Universitat Autònoma de Barcelona

2019/2020

Fundamentals of Programming

Code: 102154
ECTS Credits: 6

Degree	Type	Year	Semester
2501232 Business and Information Technology	FB	1	2

Contact

Name: Roger Malet Munté
Email: Roger.Malet@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

Marc Vallribera Ros

Prerequisites

To be able to correctly follow this subject, some previous algorithmic knowledge and problem solving are required. This knowledge is achieved through the subject "Introduction to Problem Solving and Algorithm Design" (102151), a first year, first semester course which gives the necessary mechanisms to deal with the problems that must be resolved through programming in a high level language.

Objectives and Contextualisation

The goals of this subject are to introduce the basic concepts of the programming of computer applications to solve problems that can occur in the business field. The idea of this course is to be able to implement algorithms in a programming language in order to be able to generate final solutions that solve specific problems and automate certain computer processes that occur in the companies. In addition, understanding the process of generating applications must allow a greater understanding of the more complex computer processes that occur in this same field.

Thus, the training objectives of the subject can be summarized in:

- Understand and use the basic structures of structured programming

- Implement algorithms in a high-level programming language

Competences

- Demonstrating a concern for quality in the objectives and development of the work.
- Finding algorithmic solutions and using the appropriate programming tools in order to implement them in a organization environment.
- Students must be capable of analysing, summarising, organising, planning and solving problems and making decisions.
- Working in teams, sharing knowledge and communicating it to the rest of the team and the organisation.

Learning Outcomes

1. Create a more efficient algorithm in order to solve a problem.
2. Demonstrating a concern for quality in the objectives and development of the work.
3. Encoding an algorithm using the structures of structured programming.
4. Proposing solutions based on object-oriented programming.
5. Students must be capable of analysing, summarising, organising, planning and solving problems and making decisions.
6. Using the basic structures of structured programming.
7. Working in teams, sharing knowledge and communicating it to the rest of the team and the organisation.

Content

Topic 1: Introduction

- Goal:

Introduce the way computers process information, the role of programming languages and the process of developing applications.

- Contents:

1.1. Computers and algorithms: Information processing

1.2. Programming languages and programs

1.3. Software development with the computer

1.4. Storing information on the computer

Topic 2: Introduction to C# programming

- Goal:

First contact with a programming language, and its main features.

- Contents:

2.1. My first program

2.2. Elements of the programming language

2.3. The data in the programming language

2.4. The variables

2.5. Working with data

2.6. Input/output

2.7. Examples

Topic 3: Control structures

- Goal:

Introduction to structures that allow a program to make decisions and apply basic algorithms.

- Contents:

3.1. General structure of a program: control structures

3.2. Structures sequence

3.3. Selection structures

3.4. Iteration structures

3.5. More about the increment / decrement operator

3.6. Examples

Topic 4: Structured programming

- Goal:

Introduction to the structures that allow to organize the code so that the applications are readable and scalable.

- Contents:

4.1. Structured and modular programming

4.2. Procedures and functions

4.3. Functions

4.4. Procedures

4.5. Variables and i parameters

4.6. Recursiveness

4.7. Examples

Topic 5: Vectors (arrays): Lists and tables

- Goal:

Study of the elements that allow to create lists and tables of elements in C#.

- Contents:

5.1. Introduction

5.2. Vectors (lists)

5.3. Chains (lists) of characters

5.4. Matrices

5.5. Parameter passing

5.6. Operating with vectors (lists)

5.7. Examples

Topic 6: Structures

- Goal:

Introduction to the types of data defined by the programmer, which help to create more complex types than those defined by the C# language itself.

- Contents:

6.1. Introduction to structures (registries)

6.2. Nested structures

6.3. Structures passing

6.4. Examples

Topic 7: files

- Goal:

Introduction to the mechanisms to work with files in C#.

- Contents:

7.1. Introduction

7.2. Writing files

7.3. Reading files

7.4. Read/write functions

7.5. Random access files

7.6. Parameter passing on the argument

Topic 8: Object Oriented (O.O.) programming introduction

- Goal:

Introduction to the programming paradigm that sees a program as a series of objects or entities that are related to each other.

- Contents:

8.1. What is O.O. programming?

8.2. O.O. programming characteristics.

8.3. Class and instance.

8.4. Examples

Methodology

The teaching methodology will be based on three types of activity:

- Guided activity: theoretical classes, practices and problem analysis
- Supervised activity: attendance to tutorials and execution of executions with scheduled follow-up

- Autonomous activity: student study and case resolution, individually or in groups

Activities

Title	Hours	ECTS	Learning Outcomes
Type: Directed			
Laboratory sessions	10	0.4	3, 1, 4, 5, 7, 6
Problems classes	10	0.4	3, 1, 4, 5, 7, 6
Theoretical classes	28	1.12	3, 1, 4, 5, 6
Type: Supervised			
Personal tutorials	15	0.6	3, 1, 4, 5, 6
Type: Autonomous			
Preparation and resolution of exercises and practices	83	3.32	3, 1, 4, 5, 7, 6

Assessment

The assessment of the course will be carried out continuously, through exams, problems related to the theory classes and specific work of the laboratory sessions. The weight of each of the components will be:

- 30% for the practices mark,
- 10% the exercises proposed in class individually or by groups,
- 25% the first exam, which will take place during the middle of the semester, and
- 35% the second exam, which will be carried out at the end of the semester, on the date fixed by the Faculty. Students who have obtained a minimum mark of 5 in the first exam may opt for an additional substitute test in the computer room, on the date that will be established in a timely manner and that will coincide with the final class week.

The student will pass the subject if he or she obtains a minimum grade of 5.0 in the weighted average of the marks obtained in the activities described above. In order to be able to make the average, the following conditions must be met:

- All practices must be provided, with a minimal mark of 5.
- The weighted average of the exams must exceed 4.

Students who have obtained a mark higher than 3.5 but who do not reach 5 will have the right to a recovery, in the terms described below (retake process).

A student will be considered "non-evaluable" in the subject when he or she has not participated in any of the assessment activities.

Calendar of evaluation activities

The dates of the evaluation activities (exercises, assignments ...) will be announced well in advance during the semester.

The dates of the midterm and final exams are scheduled in the assessment calendar of the Faculty.

"The dates of evaluation activities cannot be modified, unless there is an exceptional and duly justified reason why an evaluation activity cannot be carried out. In this case, the degree coordinator willcontactboth the

teaching staff and the affected student, and a new date will be scheduled within the same academic period to make up for the missed evaluation activity." ***Section 1 of Article 115. Calendar of evaluation activities (Academic Regulations UAB).*** Students of the Faculty of Economics and Business, who in accordance with the previous paragraph need to change an evaluation activity date must process the request by filling out an Application for exams' reschedule at https://eformularis.uab.cat/group/deganat_feie/application-for-exams-reschedule

Grade revision process

After all grading activities have ended, students will be informed of the date and way in which the course grades will be published. Students will be also be informed of the procedure, place, date and time of grade revision following University regulations.

Retake Process

"To be eligible to participate in the retake process, it is required for students to have been previously been evaluated for at least two thirds of the total evaluation activities of the subject." Section 3 of Article 112 ter. The recovery (UAB Academic Regulations). Additionally, it is required that the student to have achieved an average grade of the subject between 3.5 and 4.9.

The date of the retake exam is posted in the calendar of evaluation activities of the Faculty. Students taking this exam and passing will get a grade of 5 for the subject. For the students that do not pass the retake, the grade will remain unchanged, and hence, will fail the course.

Irregularities in evaluation activities

Despite other disciplinary measures deemed appropriate, and in accordance with current academic regulations, *"whenever a student makes any irregularity that could lead to a significant variation in the grade of an evaluation activity, it will be graded with a 0, regardless of the disciplinary process that can be instructed. In case of occurrence of various irregularities in the evaluation of the same subject, the final grade of this subject will be 0"*. ***Section 10 of Article 116. Results of the evaluation. (UAB Academic Regulations).***

Assessment Activities

Title	Weighting	Hours	ECTS	Learning Outcomes
Class exercises	10	0	0	3, 2, 1, 4, 5, 7, 6
Delivery of practices	30	0	0	3, 1, 4, 5, 7, 6
First exam	25	2	0.08	3, 1, 6
Segon examen	35	2	0.08	3, 1, 4, 5, 6

Bibliography

"La Bíblia de C #", Ferguson, B. Patterson, J. Beres, P. Boutquin, M. Gupta, Anaya Multimedia

Algorithms + Data Structures = Programs, Niklaus Wirth, Prentice-Hall series in automatic computation, 1975

Online reference of C#: http://www.csharpya.com.ar/;
https://msdn.microsoft.com/en-us/library/aa288436(v=vs.71).aspx