



Universitat de Lleida

DEGREE CURRICULUM
**INTRODUCTION TO
PROGRAMMING I**

Coordination: ALSINET BERNADO, MARIA TERESA

Academic year 2023-24

Subject's general information

Subject name	INTRODUCTION TO PROGRAMMING I			
Code	102000			
Semester	1st Q(SEMESTER) CONTINUED EVALUATION			
Typology	Degree	Course	Character	Modality
	Bachelor's Degree in Computer Engineering	1	COMMON/CORE	Attendance-based
	Double bachelor's degree: Degree in Computer Engineering and Degree in Business Administration and Management	1	COMMON/CORE	Attendance-based
	Programa Acadèmic de Recorregut Successiu - Enginyeria Informàtica	1	COMMON/CORE	Attendance-based
Course number of credits (ECTS)	6			
Type of activity, credits, and groups	Activity type	PRALAB		TEORIA
	Number of credits	3		3
	Number of groups	5		2
Coordination	ALSINET BERNADO, MARIA TERESA			
Department	COMPUTER ENGINEERING AND DIGITAL DESIGN			
Teaching load distribution between lectures and independent student work	GEI: 6 ECTS = 25x6 = 150 horas de trabajo 40% --> 60 horas presenciales 60% --> 90 horas de trabajo autónomo del estudiante			
Important information on data processing	Consult this link for more information.			
Language	Catalan			

Teaching staff	E-mail addresses	Credits taught by teacher	Office and hour of attention
ALSINET BERNADO, MARIA TERESA	teresa.alsinet@udl.cat	9	EPS Office 2.13 Send an e-mail to the teacher to establish the hour of attention convenient for the student.
ARGELICH ROMA, JOSEP	josep.argelich@udl.cat	12	EPS Office 2.17 Send an e-mail to the teacher to establish the hour of attention convenient for the student.

Subject's extra information

To address the subject is advisable to show interest in analyzing real problems and developing technological solutions to solve them. It is also advisable to show analytical skills, logical reasoning and critical capacity.

This subject is scheduled in the fall semester of the 1st year.

The knowledge and competencies acquired in this subjects will be useful to follow other subjects with contents related with programming languages, data structure and algorithms.

Learning objectives

The learning objectives of the course are knowing how to apply sequential algorithm analysis and design techniques to later implement them in a high-level programming language and solve real data processing problems. Specifically, the programming language used for this purpose is ANSI C/C++ and problems to be solved are mainly those related with sequences processing.

In particular the main student's learning objectives are:

- To design and implement algorithmic structures to solve the different types of problems.
- To design and implement data structures to encode information.
- To design and implement iterative algorithms.
- To identify problem types and to apply appropriate algorithmic strategies.
- To design and implement algorithms to solve complex problems in a structured way.
- To design and implement solutions using the top-down design technique.
- To use a software development environment based on a high-level programming language.

Competences

Strategic Competences of the UdL:

- EPS1. Capacity to solve problems and prepare and defence arguments inside the area of studies.
- EPS5. Capacity of abstraction and of critical, logical and mathematical thinking.
- EPS9. Capacity for unidisciplinary and multidisciplinary teamwork.
- EPS12. To be motivated for the quality and steady improvement.

Specific competences in the degree in Computer Engineering:

- GII-FB3. Capacity to understand and master the basic concepts of discreet mathematics, logical, algorithmic and computational complexity, and its application to solve engineering problems.
- GII-FB4. Basic knowledge of the use and programming of computers, operating systems, databases and

computer programs with applications in engineering.

- GII-FB5. Knowledge of the structure, organisation, operation and interconnection of the computer systems, the basics of programming, and its application to solve engineering problems.
- GII-CRI7. Knowledge, design and efficient use of the types and data structure more suitable for solving a problem.
- GII-CRI9. Capacity to know, comprise and evaluate the structure and architecture of computers, as well as the basic components that conform them.

Subject contents

Introduction: Processes, algorithms and programs.

Unit 1. Basic algorithmic structures

1.1 Constants, variables, basic types and valid expressions

1.2 Assignment, sequential composition, alternative composition and iterative composition

1.3 Programming Environment

Unit 2. Iterative design of programs

2.1 Sequential Access

- Algorithmic schemes for sequence processing
- Algorithmic schemes for search in sequences

2.2 Direct access. Tables

- Sequential tables
- Direct tables
- Multidimensional tables
- Classic sorting algorithms

Unit 3. Non-basic data processing

3.1 Procedures and Functions

3.2 Parameter transfer mechanisms

3.3 Descendant design of algorithms

Methodology

Each week students attend 2 hours with a Large group and 2 hours with a Medium Group. Medium Group sessions are taught in the laboratory.

Large Groups: Theory and Problems (3 ECTS)

- Theory: classes supported with slides and/or notes.
- Part of practical application: always work with problems and programming exercises.

Medium Groups: Laboratory (3 ECTS)

- Tutorials and personalized follow-up for practice groups. The teacher provides a collection of problems. Solutions are developed along the semester.
- Using compilers and editing tools.
- Continuous work driven by means of two mandatory practices.

Autonomous work:

- The practice will be completed with non-contact hours. In the Medium Group sessions the teacher supports mandatory practices which must be developed by the student throughout the course autonomously.
- It is recommended that students solve all problems from the collection problem, in order to practice and get feedback from the teacher.

Development plan

Sem	Descripción	Actividad Presencial GG	Actividad Presencial GM	Trabajo autónomo
1	Presentation Introduction	Introduction to the course. Introduction: Processes, algorithms and programs	Using a programming environment	To solve programming exercises
2	Basic algorithmic structures	Unit 1. Constants, variables, basic types and valid expressions	Programming Exercises	To solve programming exercises
3	Basic algorithmic structures	Unit 1. Assignment, sequential composition and alternative composition	Programming Exercises	To solve programming exercises
4	Basic algorithmic structures	Unit 1. The iterative composition	Programming Exercises	To solve programming exercises
5	Iterative design of programs	Unit 2. Sequential Access	Practice 1: Overview of the first mandatory practice	To implement Practice 1 in groups
6	Iterative design of programs	Unit 2. Search in sequences	Programming Exercises Support for Practice 1	To solve programming exercises To implement Practice 1 in groups
7	Iterative design of programs	Unit 2. Direct access. Tables	Programming Exercises Support for Practice 1	To solve programming exercises To implement Practice 1 in groups
8	Iterative design of programs	Unit 2. Programming Exercises with tables: treatment and search.	Programming Exercises Support for Practice 1	To solve programming exercises To implement Practice 1 in groups
9		1st Assessment	Delivery Practice 1	To study To implement Practice 1 in groups
10	Iterative design of programs	Unit 2. Multidimensional tables	Classic sorting algorithms	To solve programming exercises

11	Non-basic data processing	Unit 3. Procedures and Functions	Programming Exercises	To solve programming exercises
12	Non-basic data processing	Unit 3. Parameter transfer mechanisms	Practice 2: Overview of the second mandatory practice	To implement Practice 2 in groups
13	Non-basic data processing	Unit 3. Descendant design of algorithms	Programming Exercises Support for Practice 2	To solve programming exercises To implement Practice 2 in groups
14	Non-basic data processing	Unit 3. Programming exercises: Descendant design of algorithms	Programming Exercises Support for Practice 2	To solve exercises To implement Practice 2 in groups
15	Non-basic data processing	Unit 3. Programming exercises: Descendant design of algorithms	Programming Exercises Support for Practice 2	To solve programming exercises To implement Practice 2 in groups
16		2nd Assessment	Delivery Practice 2	To study To implement Practice 2 in groups
17		2nd Assessment		To study
18				
19		Improvement of the 2nd assessment		To study

Evaluation

The continuous evaluation of the subject is based on 3 blocks:

- **Practice Block: 25%.** It consists of two activities: Practice 1 and Practice 2. They cannot be improved. A minimum grade is not required. The delivery date of Practice 1: the date of the realization of the 1st Assessment, defined by the EPS. The delivery date of Practice 2: the date of the realization of the 2nd Assessment, defined by the EPS.
- **Theory block 1: 25%.** It consists of one activity: 1st *Assessment*. It can be improved with the theory block 2. A minimum grade is not required. The date of the exam: the date of the realization of the 1st *Assessment*, defined by the EPS.
- **Theory block 2: 50%.** It consists of one activity: 2nd *Assessment*. It can be improved with the **Improvement of Theory Block 2**. A minimum grade is not required. The date of the exam: the date of the realization of the 2nd *Assessment*, defined by the EPS.

Improvement of Theory Block 2: It consists of performing the 2nd Assessment again. A minimum grade is not required. The date of the exam: the date of the realization of the Improvement of the 2nd Assessment, defined by the EPS. The realization of Improvement of Theory Block 2 does not condition the maximum grade achieved in the subject.

Evaluation activities

Acronym	Evaluation activities	Weighing	Minimum grade required	Team work	Mandatory	Improvement
P1	Assessment 1	25%	NO	NO	NO	YES
P2	Assessment 2	50%	NO	NO	NO	YES
PRA1	Practice 1	10%	NO	Maximum 2 people	NO	NO
PRA2	Practice 2	15%	NO	Maximum 2 people	NO	NO
To pass the subject, the final grade must be ≥ 5 .						
Final grade = $0,25 \cdot P1 + 0,5 \cdot P2 + 0,1 \cdot PRA1 + 0,15 \cdot PRA2$						

Remarks:

- If the grade obtained in the written test P2 is ≥ 5 , then this grade may act as an improvement of the first written test P1, the weight of which is 25%.
- The student can choose to improve the written test P2, whose weight is 50%, by taking the corresponding improvement of Theory Block 2 on the date defined by the EPS. The realization of the improvement of Theory Block 2 does not condition the maximum grade achieved in the subject.

Alternative assessment (students who waive continuous assessment):

- **Practice 1: 10%**. It cannot be improved. A minimum grade is not required. The delivery date of Practice 1: the date of the realization of the 1st Assessment, defined by the EPS.
- **Practice 2: 15%**. It cannot be improved. A minimum grade is not required. The delivery date of Practice 2: the date of the realization of the 2nd Assessment, defined by the EPS.
- **Assessment 2: 75%**. It can be improved with the Improvement of Assessment 2. A minimum grade is not required. The date of the exam: the date of the realization of the 2nd *Assessment*, defined by the EPS.
- **Improvement of Assessment 2: 75%**. It consists of performing the 2nd Assessment again. A minimum grade is not required. The date of the exam: the date of the realization of the Improvement of the 2nd Assessment, defined by the EPS. The realization of Improvement of Assessment 2 does not condition the maximum grade achieved in the subject.

Bibliography

Basic References:

- T. Alsinet, J. Argelich and S. Vila. Programació I: Notes de curs. Eines, Edicions de la Universitat de Lleida, col.lecció UdL Eines 81, 2017.
- J. Castro, F. Cucker, X. Messeguer, A. Rubio, L. Solano and B.Valles. *Curs de Programació*. McGraw-Hill, 1992.
- G. Brassard and P. Bratley. *Fundamentos de Algoritmia*. Prentice Hall, 1997.
- L. Joyanes. Fundamentos de Programación. Algoritmos, Estructuras de Datos y Objetos. McGraw-Hill, 2003.

ANSI C and C++:

- H.M. Deitel and P.J. Deitel. *Como Programar en C/C++*. Prentice-Hall, segunda edición, 2002.
- B. Stroustrup. Programming -- Principles and Practice Using C++. Addison Wesley, 2008.
- L. Joyanes. Programación en C++. McGraw-Hill, 2006.