# 1. OVERVIEW

| | |
|---|---|
| Subject area | Data Structure and Algorithms |
| Degree | Bachelor's Degree in Data Science |
| School/Faculty | Faculty of Science, Engineering and Design |
| Year | Second |
| ECTS | 6 ECTS |
| Type | Core |
| Language(s) | Spanish |
| Delivery Mode | On campus |
| Semester | Second semester |

# 2. INTRODUCTION

Correct use of linear and non-linear programming allows us to offer services for dealing with large volumes of data. Queues, trees and graphs are used in most distributed processing environments. Therefore, this subject aims to teach students advanced techniques for design, development, monitoring and testing, as well as how to solve data extraction, transformation and load problems on a business level. This, all the while, following code quality and good practices.

We will also introduce cases where the student can effectively include high-quality code outlining methods with a consistent focus on data management and processing. This subject will give students an overall view of data programming which will serve as a base for other subjects in the Bachelor's Degree in Data Science.

# 3. SKILLS AND LEARNING OUTCOMES

**Basic skills (CB, by the acronym in Spanish):**

- CB1: Students have shown their knowledge and understanding of a study area originating from general secondary school education, and are usually at the level where, with the support of more advanced textbooks, they may also demonstrate awareness of the latest developments in their field of study.
- CB2: Students can apply their knowledge to their work or vocation in a professional manner and possess the skills which are usually evident through the forming and defending of opinions and resolving problems within their study area

**Cross-curricular skills (CT, by the acronym in Spanish):**
- CT02: Independent learning: skills for choosing strategies to search, analyse, evaluate and manage information from different sources, as well as to independently learn and put into practice what has been learnt.

- CT04. Written/spoken communication: ability to communicate and gather information, ideas, opinions and viewpoints to understand and be able to act, spoken through words or gestures or written through words and/or graphic elements.
- CT05: Analysis and problem-solving: be able to critically assess information, break down complex situations, identify patterns and consider different alternatives, approaches and perspectives in order to find the best solutions and effective negotiations.

**Specific skills (CE, by the acronym in Spanish):**

- CE3. Knowledge of the core principles and applications of software development and databases.
- CE4. Ability to successfully apply data type models and algorithms to create solutions to problems in the data science field.
- CE5. Ability to design, implement, gather, store and exploit databases and database management systems to create solutions to problems in the data science field.

**Learning outcomes (RA, by the acronym in Spanish):**

- Develop algorithmic thinking. Transfer a problem into a sequence of actions to solve.
- Design and implement solutions to problems of medium-level complexity using databases (structured, semi-structured, non-structured), data structures and object-oriented programming.
- Use programming environments to compile, link and execute programs, as well as identify and correct errors in each stage.
- Suitably document the designs, as well as the introduction of comments in the code to ease understanding and further use of the software created.

The following table shows how the skills developed in the subject area match up with the intended learning outcomes:

| Skills | Learning outcomes |
|---|---|
| CB1, CB2, CT05, CE3, CE4 | Develop algorithmic thinking. Transfer a problem into a sequence of actions to solve. |
| CE4, CE5 | Design and implement solutions to problems of medium-level complexity using databases (structured, semi-structured, non-structured), data structures and object-oriented programming. |
| CE5 | Use programming environments to compile, link and execute programs, as well as identify and correct errors in each stage. |
| CT02, CT04, | Suitably document the designs, as well as the introduction of comments in the code to ease understanding and further use of the software created. |

# 4. CONTENTS

Data Structure and Algorithms
1. Linear data structure
2. Hierarchical data structure
3. Lists, queues and stacks
4. Trees and graphs
5. Algorithmic schemes

# 5. TEACHING/LEARNING METHODS

The types of teaching/learning methods are as follows:

• Collaborative learning: students learn to work with other people (colleagues and professors) to find creative, comprehensive and constructive solutions to questions and problems that arise from the given case studies, using relevant knowledge and available resources in relation to each subject.
• Problem-based learning: students face problems they must solve either working as a team or independently.
• Master Lecture: presentations by the professor using the appropriate technological tools to facilitate understanding of the subject matter.
• Directed academic activities: more independent tasks (individual or in groups), involving search for information, written summaries, debates and public defence of work.

# 6. LEARNING ACTIVITIES

The types of learning activities, plus the amount of time spent on each activity, are as follows:

**On campus:**

| Learning activity | Number of hours |
|---|---|
| Master classes | 44 |
| Problem solving and case studies | 10 |
| Laboratory work (exercises led by the teacher) | 15 |
| Knowledge tests | 4 |
| Independent working | 50 |
| Tutorials | 6 |
| Practical work at home | 21 |
| TOTAL | 150 |

**Online:**

| Learning activity | Number of hours |
|---|---|
| Online classes with presentations | 44 |
| Online classes with problem-solving and case studies | 10 |
| Computer practice at home (exercises led by the teacher) | 15 |
| Knowledge tests | 4 |

| | |
|---|---|
| Independent working | 50 |
| Tutorials | 6 |
| Practical work at home | 21 |
| TOTAL | 150 |

# 7. ASSESSMENT

The assessment methods, plus their weighting in the final grade for the subject area, are as follows:

**On campus:**

| Assessment system | Weighting |
|---|---|
| On campus tests to evaluate objectives of theory/practical learning (exam-type objective tests, written compositions, spoken presentations, case studies/problem solving, debates, simulation tests) | 40% |
| On campus laboratory tests (activity reports, spoken presentations) | 25% |
| Off-site tests to assess theory/practical learning (case studies/problem solving) | 25% |
| Attitude assessment tests (attitude assessment rules, class participation) | 5% |
| Self- and co-assessment (learning contract, learning outcomes) | 5% |
| TOTAL | 100% |

**Online:**

| Assessment system | Weighting |
|---|---|
| Online tests to evaluate objectives of theory/practical learning (exam-type objective tests, written compositions, spoken presentations, case studies/problem solving, debates, simulation tests) | 40% |
| Online laboratory tests (activity reports, spoken presentations) | 25% |
| Off-site tests to assess theory/practical learning (case studies/problem solving) | 25% |
| Attitude assessment tests (attitude assessment rules, class participation) | 5% |
| Self- and co-assessment (learning contract, learning outcomes) | 5% |
| TOTAL | 100% |

On the Virtual Campus, when you open the subject area, you can see all the details of your assessment activities and the deadlines and assessment procedures for each activity.

# 8. BIBLIOGRAPHY

The reference publication to accompany this subject area is:

- Joyanes, L. (2008). Fundamentos de la programación. Algoritmos y Estructura de Datos, 4ª Edición. Madrid: McGraw-Hill.

The recommended bibliography is indicated below:

• Brookshear, J. G. (2012). Computer Science: An Overview. Massachusetts: Pearson/Addison-Wesley.

• Cerrada, J. A. y Collado, M. E. (2010). Fundamentos de Programación. Madrid: Editorial Uni-versitaria Ramón Areces.

• Dixit, J. B. (2009). Computer Fundamentals and Programming. New Delhi: Laxmi Publications

• Guardati S Buemo (2020), De cero al infinito, Aprende a programar en Python

• Jiménez, M. y Otero, B. (2013). Fundamentos de ordenadores: programación en C. Barcelona: Editorial Universitat Politècnica de Catalunya.

•Joyanes, L. (2008). Fundamentos de la programación. Algoritmos y Estructura de Datos, 4ª Edición. Madrid: McGraw-Hill.

• Juganaru, M. (2014). Introducción a la programación. Ciudad de México: Grupo Editorial Pa-tria.

• Moreno Alfredo (2021) Python Avanzado.

• Peña, R. (2005). Diseño de Programas: Formalismo y Abstracción. Madrid: Pearson.

• Rodríguez, M. A. (1991). Metodología de programación a través de pseudocódigo. Madrid: McGraw-Hill.

• Romney, M. B. y Steinbart, P. J. (2011). Accounting Information Systems. New York: Prentice Hall.

• Sympsom Olivier (2020) Python Data science for Begineers