# Course guide
# 270086 - CAP - Advanced Programming Concepts

**Last modified:** 11/07/2025

**Unit in charge:** Barcelona School of Informatics
**Teaching unit:** 723 - CS - Department of Computer Science.

**Degree:** BACHELOR'S DEGREE IN INFORMATICS ENGINEERING (Syllabus 2010). (Optional subject).

**Academic year:** 2025 **ECTS Credits:** 6.0 **Languages:** Catalan

## LECTURER

**Coordinating lecturer:** GERARD ESCUDERO BAKX - JORDI DELGADO PIN

**Others:** Primer quadrimestre:
JORDI DELGADO PIN - 11, 12, 13
GERARD ESCUDERO BAKX - 11, 13

## PRIOR SKILLS

Students should have enough knowledge of data structures and algorithms

## REQUIREMENTS

- Prerequisite IES
- Prerequisite PROP

## DEGREE COMPETENCES TO WHICH THE SUBJECT CONTRIBUTES

**Specific:**
CCO2.2. Capacity to acquire, obtain, formalize and represent human knowledge in a computable way to solve problems through a computer system in any applicable field, in particular in the fields related to computation, perception and operation in intelligent environments.
CES1.1. To develop, maintain and evaluate complex and/or critical software systems and services.
CES1.3. To identify, evaluate and manage potential risks related to software building which could arise.
CES1.7. To control the quality and design tests in the software production
CT1.2C. To use properly theories, procedures and tools in the professional development of the informatics engineering in all its fields (specification, design, implementation, deployment and products evaluation) demonstrating the comprehension of the adopted compromises in the design decisions.
CT4.1. To identify the most adequate algorithmic solutions to solve medium difficulty problems.
CT4.2. To reason about the correction and efficiency of an algorithmic solution.
CT5.1. To choose, combine and exploit different programming paradigms, at the moment of building software, taking into account criteria like ease of development, efficiency, portability and maintainability.
CT5.2. To know, design and use efficiently the most adequate data types and data structures to solve a problem.
CT5.3. To design, write, test, refine, document and maintain code in an high level programming language to solve programming problems applying algorithmic schemas and using data structures.

**Generical:**
G5. TEAMWORK: to be capable to work as a team member, being just one more member or performing management tasks, with the finality of contributing to develop projects in a pragmatic way and with responsibility sense; to assume compromises taking into account the available resources.

## TEACHING METHODOLOGY

Teaching the course is structured in lectures and laboratory sessions.

Teachers will use lectures to introduce the essential contents of the course. In the laboratory sessions the contents of the course will be brought to the computer by carrying out practical problems. The laboratory classes will be a continuation of the lectures, where new concepts will be implemented as they appear in lectures.

## LEARNING OBJECTIVES OF THE SUBJECT

1.Learn the capabilities provided to a programming language by having higher-order functions
2.Learn the techniques associated with higher-order functions in the Clojure programming language
3.Knowing what a closure is and some techniques associated with their use.
4.Learn a dynamic and functional programming language such as Clojure.
5.Learn immutable data structures and the consequences of having them in a programming language.
6.Learn the concept of lazy evaluation and the consequences of being able to choose between different types of evaluation
7.Being able to develop lab exercises that uses some of the taught material

## STUDY LOAD

| Type | Hours | Percentage |
|---|---|---|
| Hours large group | 30,0 | 20.00 |
| Hours small group | 30,0 | 20.00 |
| Guided activities | 6,0 | 4.00 |
| Self study | 84,0 | 56.00 |

**Total learning time:** 150 h

## CONTENTS

### Clojure introduction

**Description:**
Expressions, conditionals, local variables (let), definition of functions, sequences (lists, vectors, strings), recursion and iteration.

### First-class Functions

**Description:**
Functions as parameters, storing functions in data structures. Ways to iterate over collections: reduce, map, filter, foldr, etc. Functions that return functions

### Closures: The environment model

**Description:**
Lexical scope, The functions are not such: They capture the lexical context, If we have closures, no objects are needed: Examples. Explain how closures work using the environment model.

**Programming with higher order functions: Techniques.**

**Description:**
Composition, Pipelining, CPS, TCO & Trampolining

**Immutable data structures**

**Description:**
Advantages and disadvantages. The case of Clojure: Separate state management and make it explicit (immutability by default)

**Lazy sequences**

**Description:**
"Infinite" data structures

**Macros**

**Description:**
The concept of 'Macro' in the world of Lisp languages. Reflection

## ACTIVITIES

**Clojure introduction**

**Description:**
The student should pay attention to the lecture and he/she should work through the exercises suggested by the lecturer.

**Specific objectives:**
4

**Full-or-part-time:** 18h
Self study: 10h
Theory classes: 4h
Laboratory classes: 4h

**First-class Functions and Closures: The environment model**

**Description:**
The student should pay attention to the lecture and he/she should work through the exercises suggested by the lecturer.

**Specific objectives:**
1, 2, 3

**Full-or-part-time:** 34h
Self study: 18h
Theory classes: 8h
Laboratory classes: 8h

**Programming with higher order functions: Techniques.**

**Description:**
The student should pay attention to the lecture and he/she should work through the exercises suggested by the lecturer.

**Specific objectives:**
1, 2, 3

**Full-or-part-time:** 34h
Self study: 18h
Theory classes: 8h
Laboratory classes: 8h

**Theory test**

**Specific objectives:**
1, 2, 3, 4

**Full-or-part-time:** 14h
Self study: 12h
Guided activities: 2h

**Immutable data structures**

**Description:**
The student should pay attention to the lecture and he/she should work through the exercises suggested by the lecturer.

**Specific objectives:**
5

**Full-or-part-time:** 10h
Self study: 6h
Theory classes: 2h
Laboratory classes: 2h

**Lazy sequences**

**Description:**
The student should pay attention to the lecture and he/she should work through the exercises suggested by the lecturer.

**Specific objectives:**
6

**Full-or-part-time:** 10h
Self study: 6h
Theory classes: 2h
Laboratory classes: 2h

### Macros

**Description:**
The student should pay attention to the lecture and he/she should work through the exercises suggested by the lecturer.

**Specific objectives:**
4

**Full-or-part-time:** 18h
Self study: 10h
Theory classes: 4h
Laboratory classes: 4h

### Practical exercise

**Description:**
Delivery date: Throughout the course

**Specific objectives:**
7

**Related competencies :**
G5. TEAMWORK: to be capable to work as a team member, being just one more member or performing management tasks, with the finality of contributing to develop projects in a pragmatic way and with responsibility sense; to assume compromises taking into account the available resources.

### Final test

**Specific objectives:**
1, 2, 3, 4, 5, 6

**Full-or-part-time:** 12h
Self study: 10h
Guided activities: 2h

## GRADING SYSTEM

Grading the course will consist of two theoretical tests (P and F), one in the middle of the course and the other at the end, and laboratory exercises that will be delivered in groups of two people (L). The laboratory grade will be computed as follows: Students will deliver between 5 and 7 exercises throughout the course in groups of two, and the laboratory grade will be the average of these grades.

Then, the evaluation method is = Theory*0.8 + L*0.2
where:
Theory: MAX(F,(P+F)/2)

Teamwork:

Evaluated using a simple rubric that each group tutor group uses
to rank different aspects of teamwork of every member of the group.

## BIBLIOGRAPHY

**Basic:**
- Fogus, Michael; Houser, Chris. The Joy of Clojure. 2nd edition. Shelter Island: Manning, 2014. ISBN 9781638351283.
- Emerick, Chas; Carper, Brian; Grand, Christophe. Clojure Programming: Practical Lisp for the Java World. O'Reilly Media, 2012. ISBN 9781449394707.
- Jones, Colin. Mastering Clojure Macros: Write Cleaner, Faster, Smarter Code. The Pragmatic Programmers, 2014. ISBN 1941222226.

**Complementary:**
- Reade, Chris. Elements of functional programming. Wokingham [etc.]: Addison-Wesley, 1989. ISBN 0201129159.
- Bird, Richard. Thinking Functionally with Haskell. Cambridge [etc.]: Cambridge University Press, 2014. ISBN 9781107452640.