# Degree competences to which the subject contributes

## Specific:
1. (ENG) Coneixements bàsics sobre l'ús i programació dels ordinadors, sistemes operatius, bases de dades i programes informàtics amb aplicació en enginyeria.

## Transversal:
3. EFFICIENT ORAL AND WRITTEN COMMUNICATION - Level 1. Planning oral communication, answering questions properly and writing straightforward texts that are spelt correctly and are grammatically coherent.

## Teaching methodology
- Face-to-face lecture sessions.
- Face-to-face practical work sessions.
- Independent learning and exercises.
- Preparation and completion of group activities subject to assessment.

In the face-to-face lecture sessions, the lecturer will introduce the basic theory, concepts, methods and results for the subject and use examples to facilitate students' understanding.

Practical class work will be covered in three types of sessions:

- a) Sessions in which the lecturer will provide students with guidelines to analyse data for solving problems by applying methods, concepts and theoretical results.
- d) Sessions in which students give presentations of group work.
- c) Examination sessions

Students will be expected to study in their own time so that they are familiar with concepts and are able to solve the exercises set, either manually or with the help of a computer. Students will have access to online tools during independent learning.

Students will carry out group work which will be publicly presented during application sessions.

## Learning objectives of the subject

The main aim of the Fundamentals of Informatics course is to teach students how to program using a high-level language. To pass the course, students must be able to:
320095 - FI - Foundations of Computing

- Understand the basic concepts associated with computer hardware and software, i.e. the structure of computers and operating systems.
- Understand the fundamental concepts of computer programming.
- Develop the ability to use basic programming tools and techniques: algorithms and programs.
- Design software that is well structured, efficient and readable.
- Design data structures to represent data in a given problem.
- Carry out a medium-scale industrial programming project.
- Develop the capacity for abstraction in the use of programming patterns for solving real problems.

### Study load

<table>
<thead>
<tr>
<th>Total learning time: 150h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours large group:</td>
</tr>
<tr>
<td>Hours medium group:</td>
</tr>
<tr>
<td>Hours small group:</td>
</tr>
<tr>
<td>Guided activities:</td>
</tr>
<tr>
<td>Self study:</td>
</tr>
</tbody>
</table>
# 320095 - FI - Foundations of Computing

## Content

<table>
<thead>
<tr>
<th>Topic 1: INTRODUCTION TO COMPUTERS</th>
<th>Learning time: 7h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Laboratory classes: 2h</td>
</tr>
<tr>
<td></td>
<td>Self study : 5h</td>
</tr>
</tbody>
</table>

### Description:

1.1. Computer architecture.
1.2. Operating systems.
1.3. Computer programming.
1.4. Algorithms and programs.

### Related activities:

- Activity 5, which corresponds to a group research and development project on an introductory topic.

### Specific objectives:
For students to:

- Define the terms "hardware" and "software".
- Understand the general structure of a computer.
- Understand what an operating system is.
- Name and describe the various types of operating systems.
- Know what a program is.
- Name various programming languages.
- Name and describe the various programming paradigms.
- Understand the basic requirements of a program.
- Name and describe the phases of software development.
- Understand what an algorithm is.
**Topic 2: BASICS OF STRUCTURED PROGRAMMING**

<table>
<thead>
<tr>
<th>Description</th>
<th>Learning time: 28h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Laboratory classes: 10h</td>
</tr>
<tr>
<td></td>
<td>Self study : 18h</td>
</tr>
</tbody>
</table>

**Description:**
- 2.1. Structure of a program.
- 2.2. Objects.
- 2.3. Expressions and operators.
- 2.4. Elementary actions.
- 2.5. Writing instructions.

**Related activities:**
- One type-1 activity: Individual continuous-assessment test spanning the various laboratory-group sessions.
- One or more type-2 activities: Individual take-home self-directed learning test.
- One or more type-3 activities: Individual take-home self-directed learning task.

**Specific objectives:**
- For students to:
  - Properly use the objects in a program: constants and variables.
  - Distinguish between correct and incorrect identifiers.
  - Describe and use correctly the type of data available in the programming language as well as the defined operations.
  - Describe the function of basic input and output actions and use them correctly.
  - Understand assignment statements and use them correctly.
  - Describe the structure of a program.
  - Correctly develop a test suite.
  - Determine whether the block structure of a program is correct.
  - Correctly use alternative and iterative composition.
  - Correctly build programs with objects, expressions, elementary actions and compositions.
### Topic 3: SUBROUTINES: ACTIONS AND FUNCTIONS

<table>
<thead>
<tr>
<th>Learning time: 27h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laboratory classes: 10h</td>
</tr>
<tr>
<td>Guided activities: 2h</td>
</tr>
<tr>
<td>Self study: 15h</td>
</tr>
</tbody>
</table>

#### Description:
- 3.1. Actions.
- 3.2. Functions.
- 3.3. Parameter passing.
- 3.4. Library functions.

#### Related activities:
- One type-1 activity: Individual continuous-assessment test spanning the various laboratory-group sessions.
- One or more type-2 activities: Individual take-home self-directed learning test.
- One or more type-3 activities: Individual take-home self-directed learning task.
- One phase of activity 4, the project.

#### Specific objectives:
For students to:
- Determine whether a subroutine needs to be an action or a function.
- Send parameters by value and by reference.
- Define formal and actual parameters (arguments).
- Determine whether a formal parameter of an action or function is input, output or input/output.
- Write a program that uses library functions properly.
- Implement and use functions and actions properly.
- Detect and eliminate code repetition.
- Build programs correctly with the help of functions and actions.
Topic 4: BASIC ALGORITHM

Learning time: 26h
- Laboratory classes: 10h
- Self study: 16h

Description:
4.1. Sequences.
4.2. Traversal techniques.
4.3. Search techniques.

Related activities:
- One type-1 activity: Individual continuous-assessment test spanning the various laboratory-group sessions.
- One or more type-2 activities: Individual take-home self-directed learning test.
- One or more type-3 activities: Individual take-home self-directed learning task.
- One phase of activity 4, the project.

Specific objectives:
For students to:
- Describe the concept of a sequence.
- Define the sequences associated with a problem.
- Determine, given a sequence problem, whether the scheme can be solved by a search technique or a traversal technique.
- Correctly apply traversal and search algorithms.

Topic 5: STRUCTURED TYPES

Learning time: 31h
- Laboratory classes: 14h
- Guided activities: 2h
- Self study: 15h

Description:
5.1. Tuples.
5.2. Tables.
5.3. Traversal and search techniques in tables.
5.4. Sorting and search algorithms.

Related activities:
- One type-1 activity: Individual continuous-assessment test spanning the various laboratory-group sessions.
- One or more type-2 activities: Individual take-home self-directed learning test.
- One or more type-3 activities: Individual take-home self-directed learning task.
- One phase of activity 4, the project.

Specific objectives:
For students to:
- Write the declaration of a tuple and a table.
- Correctly declare tuple and table variables and access the declarations properly.
- Correctly use pass-by-value and pass-by-reference evaluation with tuples and tables.
- Generate partially filled tables.
- Write code that inserts and removes items in a variable-length table.
- Understand and correctly use some sorting algorithms.
Table of contents:

- Topic 6: TOP-DOWN DESIGN

**Description:**
6.1. Top-down design of data structures.
6.2. Top-down design of processes.

**Related activities:**
For students to:
- Design an efficient data structure for a given problem.
- Correctly access complex data structures.
- Given a complex problem, carry out top-down design using subroutines.

**Specific objectives:**
- Completion of activity 4, the project.
Planning of activities

PROJECT

Description:
Cooperative work aimed to work with very specific aspects of the subject (activity type 4). There will be 3 sessions with 3 groups of 3 people to work items 3, 5 and 6. The activity is based on collaborative work with experts and each group at the end of each session will implement a small program about the topic.

Support materials:
Examples of projects solved.

Descriptions of the assignments due and their relation to the assessment:
The program is conducted at the end of the session. The evaluation of this project activity included within the Type 4.

Specific objectives:
At the end of the activity, the student must have achieved all the objectives of the course.

DIRECTED ACTIVITIES TEST

Description:
Cooperative work based on the sharing of a basic computer introductory unit. It will work with 3 groups of 3 components should discuss the approach to the topic and relevance of the data sources and shall make proposals for improvement. It will also work on the best way to introduce the topic publicly.

Support materials:
Wording and documentation related to the information seeking skills and oral presentation.

Descriptions of the assignments due and their relation to the assessment:
Wording and documentation related to the information seeking skills and oral presentation.

Specific objectives:
After the activity of 5 (including this one), a student must have achieved all the specific objectives of the subject 1. In activity 5 (which includes this) generic skills of information retrieval and presentation of oral work.

LABORATORY CONTROL
Qualification system

Mid-semester examination: 20%
Final examination: 30%
Laboratory sessions: 20%
Problem solving: 10%
Project: 20%

As part of the evaluation of the project it is included the evaluation of the common skills.

For those students who meet the requirements and submit to the reevaluation examination, the grade of the reevaluation exam will replace the grades of all the on-site written evaluation acts (tests, midterm and final exams) and the grades obtained during the course for lab practices, works, projects and presentations will be kept.

If the final grade after reevaluation is lower than 5.0, it will replace the initial one only if it is higher. If the final grade after reevaluation is greater or equal to 5.0, the final grade of the subject will be pass 5.0.

Regulations for carrying out activities

To pass the course, students must approve 50% of the evaluations.

Bibliography

Basic:

