220085 - Fundamentals of Programming

Coordinating unit: 205 - ESEIAAT - Terrassa School of Industrial, Aerospace and Audiovisual Engineering
Teaching unit: 723 - CS - Department of Computer Science
Academic year: 2019
Degree: BACHELOR'S DEGREE IN INDUSTRIAL TECHNOLOGY ENGINEERING (Syllabus 2010). (Teaching unit Compulsory)
ECTS credits: 6
Teaching languages: Catalan, Spanish

Teaching staff
Coordinator: MARTA GATIUS VILA
JOSEFINA LÓPEZ HERRERA
Others: MARTA GATIUS VILA - JOSEFINA LÓPEZ HERRERA - FATOS XHAFA

Degree competences to which the subject contributes

Specific:
1. A basic understanding of the use and programming of computers, operating systems, databases and computer programs with applications in engineering

Transversal:
2. 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

· Theory sessions
· Laboratory sessions
· Tasks and self-study exercises

The sessions consist of an exhibition explanation by the teacher of the contents of the content of the subject: theoretical basis, concepts, methods and results, illustrating them with examples appropriate to facilitate their understanding.

The working sessions will be practical in the classroom of two types:
- sessions in which the teacher guide students in analyzing data and solving problems by applying techniques concepts and theoretical results.
- Sessions Exam

The student will also be studied independently in order to assimilate the concepts and solving exercises (in on paper or computer). This student's work will be supported by online tools.

Learning objectives of the subject

The main goal of the subject Computer Science is to learn to develop programs using a high level language. In order to pass the subject the student has to be able to:

- Learn the basic concept related to software and hardware: computers structure and operative systems.
- Learn the basic concepts for programming.
- Develop the ability to use basic tools and basic techniques of programming: algorithms and programs.
- Learn to design programs correctly: well-structured, efficient and readable.
- Learn to design data structures representing the data involved in a given problem.
- Design and develop a medium size program.
- Develop their ability to use abstraction in programming patterns to solve real problems.

## Study load

| Total learning time: 150h | Hours large group: 0h 0.00% | Hours small group: 60h 40.00% | Self study: 90h 60.00% |
## 1: INTRODUCTION TO COMPUTERS

### 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:
- 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.

### Learning time:
- 11h
  - Laboratory classes: 2h
  - Guided activities: 0h
  - Self study: 9h
# 2. Basic programming concepts

<table>
<thead>
<tr>
<th>Description:</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1. Structure of a program.</td>
</tr>
<tr>
<td>2.2. Objects.</td>
</tr>
<tr>
<td>2.3. Expressions and operators.</td>
</tr>
<tr>
<td>2.4. Elementary actions.</td>
</tr>
<tr>
<td>2.5. Writing instructions.</td>
</tr>
</tbody>
</table>

**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:**
- 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.
- 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.
3. Subprograms: Actions and functions

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.
### 4. BASIC ALGORITHMS

**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:**
- 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.

<table>
<thead>
<tr>
<th>Learning time: 26h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laboratory classes: 10h</td>
</tr>
<tr>
<td>Self study : 16h</td>
</tr>
</tbody>
</table>

### 5. Structured type

**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:**
- 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>
<thead>
<tr>
<th>Learning time: 31h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laboratory classes: 14h</td>
</tr>
<tr>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td>Self study : 17h</td>
</tr>
</tbody>
</table>
### 6. Design descending

<table>
<thead>
<tr>
<th>Learning time: 27h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laboratory classes: 14h</td>
</tr>
<tr>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td>Self study: 13h</td>
</tr>
</tbody>
</table>

**Description:**

- 6.1. Top-down design of data structures.
- 6.2. Top-down design of processes.

**Related activities:**

- Completion of activity 4, the project.

**Specific objectives:**

- 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.
### Planning of activities

| ACTIVITY 1: LABORATORY CONTROLS | Hours: 38h  
Laboratory classes: 18h  
Self study: 20h |
|---------------------------------|----------------|
| Description:                   | Independent work in the classroom to cover all the specific learning objectives of the subject.  
Professors corrections. |
| Support materials:             | Topic notes available (PowerPoint).  
Activity statement and the official correction criteria (rubric) available. |
| Descriptions of the assignments due and their relation to the assessment: | Resolution of the exercise by the student. The activities of type 1 represent 20% of the laboratory assessment. |
| Specific objectives:           | At the end of the activity, the student must have achieved specific objectives of the subject. |

| ACTIVITY 2: INDIVIDUAL TESTS (CONTINUOUS ASSESSMENT IN ATENEA) | Hours: 2h  
Self study: 2h |
|---------------------------------------------------------------|----------------|

| ACTIVITY 3: INDIVIDUAL TASKS (CONTINUOUS ASSESSMENT IN ATENEA) | Hours: 3h  
Self study: 3h |
|---------------------------------------------------------------|----------------|

| Activity 4; midterm exam | Hours: 43h  
Laboratory classes: 18h  
Self study: 25h |
|--------------------------|----------------|

| Activity 5: Final exam   | Hours: 45h  
Laboratory classes: 20h  
Self study: 25h |
|--------------------------|----------------|

| Activity 6: Project      | Hours: 19h  
Laboratory classes: 4h  
Self study: 15h |
|--------------------------|----------------|
220085 - Fundamentals of Programming

Qualification system

- First examination: 20%
- Second examination: 30%
- Laboratory sessions: 20%
- Application/practicals: 10%
- Final Project: 20%

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

Regulations for carrying out activities

- Any of the control laboratory not performed, will be considered as non-marked
- In no case can any documentation or digital support be used in the partial or final test.

Bibliography

Basic:


Complementary:


Others resources:

- Web page of the course in atenea
- Divulgació de la programació: http://cartesius.upc.es/adminmat/programacio/desenvolupa
- Pildoras de C++: http://www.minidosis.org/#/cursos/Fl
- Jutge: https://www.jutge.org/