820006 - I - Informatics

Coordinating unit: 820 - EUETIB - Barcelona College of Industrial Engineering
Teaching unit: 723 - CS - Department of Computer Science
Academic year: 2015
Degree: BACHELOR'S DEGREE IN BIOMEDICAL ENGINEERING (Syllabus 2009). (Teaching unit Compulsory)
BACHELOR'S DEGREE IN ELECTRICAL ENGINEERING (Syllabus 2009). (Teaching unit Compulsory)
BACHELOR'S DEGREE IN CHEMICAL ENGINEERING (Syllabus 2009). (Teaching unit Compulsory)
BACHELOR'S DEGREE IN INDUSTRIAL ELECTRONICS AND AUTOMATIC CONTROL ENGINEERING (Syllabus 2009). (Teaching unit Compulsory)
BACHELOR'S DEGREE IN MECHANICAL ENGINEERING (Syllabus 2009). (Teaching unit Compulsory)
BACHELOR'S DEGREE IN ENERGY ENGINEERING (Syllabus 2009). (Teaching unit Compulsory)
ECTS credits: 6
Teaching languages: Catalan, Spanish, English

Teaching staff
Coordinator: Javier Farreres
Others: Profesorado de los departamentos LSI y ESAII.

Opening hours
Timetable: Please see notice desk.

Prior skills
None.

Requirements
No requirements.

Degree competences to which the subject contributes
Specific:
2. Understand the basics behind the use and programming of PCs, operating systems, databases and software with applications in engineering.

Transversal:
1. EFFECTIVE USE OF INFORMATION RESOURCES - Level 1. Identifying information needs. Using collections, premises and services that are available for designing and executing simple searches that are suited to the topic.

Teaching methodology
This subject consists of two weekly presental classes in a large group, and a 2-hour weekly session in the laboratory. During the large group classes, theoretical explanations will be combined with examples and active solving of exercises by the students. During the laboratory sessions the students will follow the laboratory teacher proposed activities. Autonomous learning assigned time have an importance during the term, that will be used, for example, to learn another programming language. The Programming Project will be undertaken with PBL (Project Based Learning) methodology.

Learning objectives of the subject
After passing the subject Computer Science, the student will be able to:
820006 - I - Informatics

- Define basic concepts and terms of computer science and programming.
- Write computer programs of medium complexity, with structured programming methodology and modular design.
- Learn autonomously new programming languages.
- Use computer tools that will allow to efficiently study other subjects in the next courses of his or her career, and efficiently work as a graduate engineer in future.

<table>
<thead>
<tr>
<th>Study load</th>
<th>Total learning time: 150h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours large group:</td>
<td>30h  20.00%</td>
</tr>
<tr>
<td>Hours medium group:</td>
<td>0h    0.00%</td>
</tr>
<tr>
<td>Hours small group:</td>
<td>30h   20.00%</td>
</tr>
<tr>
<td>Guided activities:</td>
<td>0h    0.00%</td>
</tr>
<tr>
<td>Self study:</td>
<td>90h   60.00%</td>
</tr>
</tbody>
</table>
### Content

<table>
<thead>
<tr>
<th>Chapter 1 - Basic concepts</th>
<th>Learning time: 8h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Theory classes: 2h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 2h</td>
</tr>
<tr>
<td></td>
<td>Self study: 4h</td>
</tr>
</tbody>
</table>

**Description:**
- Computer architecture: von Neumann model, computer elements.
- Operative system: virtual machine, resources manager.

**Related activities:**
- Theoretical classes.
- Activity 1. Tutorial on the programming framework.

**Specific objectives:**
1. Knowing basic concepts in computer science and programming.
## Chapter 2 - Algorithms

<table>
<thead>
<tr>
<th>Learning time: 56h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory classes: 12h</td>
</tr>
<tr>
<td>Laboratory classes: 12h</td>
</tr>
<tr>
<td>Self study : 32h</td>
</tr>
</tbody>
</table>

### Description:
- Scalar types: representation, range, precision, operators.
- Expressions: computing rules, operators priority.
- Variables, constants and data types.
- Algorithm structure.
- Elementary instructions: reading, writing, assigning.
- Sequential structure.
- Alternative structure.
- Iterative structure.
- Search and journey.

### Related activities:
- Theoretical classes.
- Activity 1: Theory assessments.
- Activity 2: Algorithm programming exercises.
- Activity 3: Non-presential exercises in other languages.
- Activity 4: Programming project

### Specific objectives:
- 2. Applying basic algorithmic techniques to solve engineering problems.
- 3. Being able to code algorithms by using structured programming languages.
- 4. Using basic programming techniques and tools to develop medium complexity programs.
### Chapter 3- Actions, functions and parameters

<table>
<thead>
<tr>
<th>Learning time: 40h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory classes: 8h</td>
</tr>
<tr>
<td>Laboratory classes: 8h</td>
</tr>
<tr>
<td>Self study: 24h</td>
</tr>
</tbody>
</table>

#### Description:
- Procedures and functions.
- Parameters.
- Reutilization, modular programming, top-down design.

#### Related activities:
- Theoretical classes with examples.
- Activity 1: Assessment exercises.
- Activity 2: Laboratory exercises on modular programming.
- Activity 3: Non-presential exercises in other languages.
- Activity 4: Programming project.

#### Specific objectives:
1. Applying basic algorithmic techniques to solve engineering problems.
2. Being able to code algorithms by using structured programming languages.
3. Using basic programming techniques and tools to develop medium complexity programs.
4. Develop the ability to solve engineering problems by using programming techniques.
### Chapter 4 - Structured data types

**Description:**
Arrays, homogeneous structured types.
Records, heterogeneous structured types.
Type design.

**Related activities:**
Theoretical classes with examples.
Activity 1: Assessment exercises.
Activity 2: Laboratory exercises on structured types.
Activity 3: Non-presential exercises in other languages.
Activity 4: Programming project.

**Specific objectives:**
1. Applying basic algorithmic techniques to solve engineering problems.
2. Being able to code algorithms by using structured programming languages.
3. Developing the ability to solve engineering problems by using programming techniques.

### Chapter 5 - Advanced concepts

**Description:**
Files and data bases.
Object-oriented programming.
Event-oriented programming.
Visual programming.

**Related activities:**
Theoretical classes with examples.

**Specific objectives:**
5. Develop the ability to solve engineering problems by using programming techniques.
820006 - I - Informatics

**Qualification system**

The final note of the subject results from the following addition:

\[ FN = 0.30\ CP + 0.35\ CF + 0.10\ Final\ exercise + 0.1\ Practic\ in\ other\ language + 0.15\ Lab \]

FN: final note; CP: partial assessment exercises; CF: final assessment exercise; Lab: laboratory exercises

The note of partial assessment exercises may include exercises worked out in class during the term.

Generic competence is dealt with in its own activity. Every student will be given a separate final note on the generic competence.

**Regulations for carrying out activities**

- All activities are part of the continuous assessment model of the subject. Therefore, students repeating this subject will not be allowed to save any part of their notes for the following term.
- If a student does not hand over an activity, it will be considered as non-marked.
- Students will not be allowed to consult any kind of material during the partial and final assessment exercises.

**Bibliography**

**Basic:**


**Complementary:**