230906 - PRD - Programming and Data Structures

Coordinating unit: 230 - ETSETB - Barcelona School of Telecommunications Engineering
Teaching unit: 701 - AC - Department of Computer Architecture
Academic year: 2019
Degree: BACHELOR'S DEGREE IN ELECTRONIC ENGINEERING AND TELECOMMUNICATION (Syllabus 2018).
(Teaching unit Compulsory)
ECTS credits: 6  Teaching languages: Catalan, Spanish

Teaching staff
Coordinator: Perello Muntan, Jordi
Others: Llorente Viejo, Silvia

Prior skills
In order to follow the subject appropriately, it is strongly recommended to have passed the previous programming subject of the degree (Algorithms and Programming), as many of the programming concepts explained there will be assumed already known.

Degree competences to which the subject contributes

Basic:
CB5. (ENG) GREELEC: Que els estudiants pugin desenvolupar habilitats d'aprenentatge per emprendre estudis superiors amb un alt grau d'autonomia.

Specific:
CE1. (ENG) GREELEC: Capacitat per a la resolució dels problemes matemàtics que poguin plantejar-se a l'enginyeria. Aptitud per aplicar els coneixements sobre àlgebra lineal, geometria, geometria diferencial, càlcul diferencial i integral, equacions diferencial i en derivades parcials, mètodes numèrics, algorítmica numèrica, estadística i optimització.
(Mòdul de formació bàsica).

Generical:
CG3. (ENG) GREELEC: Coneixmetn de matèries bàsiques i tecnològies que el capacitin per a l'aprenentatge de nous mètodes i tecnologies, així com que el dotin d'una gran versatilitat per adaptar-se a noves situacions.

Transversal:
CT6. (ENG) GREELEC:APRENENTATGE AUTÒNOM: Detectar deficiències en el propi coneixement i superarles mitjançant la reflexió crítica i l'elecció de la millor actuació per ampliar coneixements.

Teaching methodology
Expository method / Lecture class
Participative lecture class
Laboratory session
Cooperative work
Autonomous work
Problem/project-based learning

Learning objectives of the subject

Subject objectives:

1. The student should be able to efficiently implement programs of moderate complexity in C programming language,
using the debugger in order to detect and fix errors occurring during the execution of the program when necessary.

2. The student should understand how the system memory is used throughout the execution of a program in C, as well as the utilization of the existing library functions for managing dynamic memory.

3. The student should be able to efficiently implement and manage basic dynamic data structures, both linear (lists, stacks, queues) and non-linear (hash tables) ones.

4. The student should understand bitwise operators in C, being able to use them in basic use cases.

Learning results:

1. The student knows the basic syntax of the C programming language, and is able to use it to efficiently implement the requested programs.

2. The student is aware of the function-based modular programming benefits in C, and is capable to implement previously specified functions using references (i.e., pointers) when necessary.

3. The student is aware of the limitations of static data structures and appreciates the benefits of dynamic data structures, knowing their basic characteristics.

4. The student is able to efficiently implement and manage dynamic data structures, both linear (lists, stacks, queues) and non-linear (hash tables) ones.

5. The student is able to employ bitwise operators for basic use cases.

### Study load

<table>
<thead>
<tr>
<th>Total learning time: 150h</th>
<th>Hours large group: 39h</th>
<th>26.00%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hours small group: 26h</td>
<td>17.33%</td>
</tr>
<tr>
<td></td>
<td>Self study: 85h</td>
<td>56.67%</td>
</tr>
</tbody>
</table>
## Content

<table>
<thead>
<tr>
<th>Section</th>
<th>Learning time:</th>
<th>Description:</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Dynamic data structures</td>
<td>48h 30m</td>
<td>Presentation of basic dynamic data structures, both linear (lists, stacks, queues) and non-linear (hash tables). Implementation of linked lists as the basic component enabling the rest of dynamic data structures that will be subsequently presented. Implementation of stacks and queues. Implementation of hash tables. Concept of hash function. Implementation of basic hash functions. Examples.</td>
</tr>
</tbody>
</table>

### Description:
- **Content**: Overview of the course content, focusing on key topics.
- **Learning time**: Breakdown of the learning time into theory classes, laboratory classes, and self-study.
- **Description**: Detailed explanation of each section, providing a comprehensive understanding of the course material.
5. Bitwise operations

Description:
Number base conversion (decimal to binary, octal and hexadecimal). Bitwise operators. Bit masks. Basic examples.

Learning time: 12h 30m
Theory classes: 4h 30m
Laboratory classes: 0h
Self study: 8h

Qualification system
Laboratory: 35% (= 60% project assignments + 40% laboratory exam)
Midterm exam: 15%
Final exam: 50%
Laboratory session attendance is compulsory: unjustified absences can impact negatively on the student's laboratory mark.

Regulations for carrying out activities
It is strictly forbidden to bring lecture notes and programmable devices (mobile phone, laptop, tablet, etc.) during the subject's midterm and final exams.

Bibliography
Basic:

Complementary: