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: 2018
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. Aptitut per aplicar els coneixements sobre àlgebra lineal, geometria, geometria diferencial, càlcul diferencial i integral, equacions diferencial i en derivades parciais, mètodes numèrics, algorítmica numèrica, estadística i optimització. (Mòdul de formació bàsica).

General:
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: APRENETATGE AUTÔNOM: Detectar deficiències en el propi coneixement i superarles mitjançant la reflexió crítica i l'elevació 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 system calls 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:</th>
<th>39h</th>
<th>26.00%</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hours small group:</td>
<td>26h</td>
<td>17.33%</td>
</tr>
<tr>
<td></td>
<td>Self study:</td>
<td>85h</td>
<td>56.67%</td>
</tr>
</tbody>
</table>
## Content

<table>
<thead>
<tr>
<th>1. Introduction to C programming language</th>
<th>Learning time: 35h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Theory classes: 9h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 8h</td>
</tr>
<tr>
<td></td>
<td>Self study: 18h</td>
</tr>
</tbody>
</table>

**Description:**

<table>
<thead>
<tr>
<th>2. Functions</th>
<th>Learning time: 35h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Theory classes: 9h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 6h</td>
</tr>
<tr>
<td></td>
<td>Self study: 20h</td>
</tr>
</tbody>
</table>

**Description:**

<table>
<thead>
<tr>
<th>3. Dynamic memory management</th>
<th>Learning time: 17h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Theory classes: 6h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 2h</td>
</tr>
<tr>
<td></td>
<td>Self study: 9h</td>
</tr>
</tbody>
</table>

**Description:**
Memory types of a C program. System calls for dynamic memory management. Basic use cases: dynamic vectors.

<table>
<thead>
<tr>
<th>4. Dynamic data structures</th>
<th>Learning time: 48h 30m</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Theory classes: 10h 30m</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 8h</td>
</tr>
<tr>
<td></td>
<td>Self study: 30h</td>
</tr>
</tbody>
</table>

**Description:**
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.
5. Bitwise operations

<table>
<thead>
<tr>
<th>Learning time: 14h 30m</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory classes: 4h 30m</td>
</tr>
<tr>
<td>Laboratory classes: 2h</td>
</tr>
<tr>
<td>Self study: 8h</td>
</tr>
</tbody>
</table>

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

Qualification system
Laboratory: 35% (= 60% project assignments + 40% laboratory final 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: