270001 - PRO1 - Programming I

Coordinating unit: 270 - FIB - Barcelona School of Informatics
Teaching unit: 723 - CS - Department of Computer Science
Academic year: 2017
Degree: BACHELOR'S DEGREE IN INFORMATICS ENGINEERING (Syllabus 2010). (Teaching unit Compulsory)
ECTS credits: 7,5
Teaching languages: Catalan, Spanish, English

Teaching staff

Coordinator:
- Jose Carmona Vargas (jcarmona@cs.upc.edu)
- Lluis Padro Cirera (padro@cs.upc.edu)

Others:
- Conrado Martinez Parra (conrado@cs.upc.edu)
- Edelmira Pasarella Sanchez (edelmira@cs.upc.edu)
- Emma Rollón Rico (erollon@cs.upc.edu)
- Jorge Castro Rabal (castro@cs.upc.edu)
- Jose Miguel Rivero Almeida (rivero@cs.upc.edu)
- Lluis Vila Grabulosa (vila@cs.upc.edu)
- Rosa Maria Jiménez Gómez (jimenez@cs.upc.edu)

Prior skills

Secondary school.

Degree competences to which the subject contributes

Specific:

CT1.1A. To demonstrate knowledge and comprehension about the fundamentals of computer usage and programming, about operating systems, databases and, in general, about computer programs applicable to the engineering.
CT1.1B. To demonstrate knowledge and comprehension about the fundamentals of computer usage and programming. Knowledge about the structure, operation and interconnection of computer systems, and about the fundamentals of its programming.
CT1.2B. To interpret, select and value concepts, theories, uses and technological developments related to computer science and its application derived from the needed fundamentals of mathematics, statistics and physics. Capacity to understand and dominate the physical and technological fundamentals of computer science: electromagnetism, waves, circuit theory, electronics and photonics and its application to solve engineering problems.
CT4.1. To identify the most adequate algorithmic solutions to solve medium difficulty problems.
CT4.2. To reason about the correction and efficiency of an algorithmic solution.
CT5.2. To know, design and use efficiently the most adequate data types and data structures to solve a problem.
CT5.3. To design, write, test, refine, document and maintain code in an high level programming language to solve programming problems applying algorithmic schemas and using data structures.
CT5.4. To design the programs' architecture using techniques of object orientation, modularization and specification and implementation of abstract data types.
CT8.6. To demonstrate the comprehension of the importance of the negotiation, effective working habits, leadership and communication skills in all the software development environments.

Generical:

G7. AUTONOMOUS LEARNING: to detect deficiencies in the own knowledge and overcome them through critical reflection and choosing the best actuation to extend this knowledge. Capacity for learning new methods and
270001 - PRO1 - Programming I

... technologies, and versatility to adapt oneself to new situations.

**Teaching methodology**

In the theory sessions, the lecturer will alternate new theoretical concepts with examples and exercises. Lectures, in which the course topics are presented, explained and illustrated, will be combined with student interaction regarding the various alternatives arising in the resolution of practical cases.

The laboratory sessions have two distinct parts: During the first hour, a guided session takes place, where the lecturer describes practical issues regarding the programming environment, or some exercises are solved in a collaborative way, or some code is analyzed to identify errors, etc. Then students devote the remaining two hours to solve problems with the automatic judge with the assistance of the lecturer if needed.

Students are expected, in the laboratory sessions and in home study, to resolve problems from a set of problems and upload their solutions to an automatic judge for checking by comprehensive test suites. They are also advised to regularly consult their lecturer about their programs (irrespective of whether they work) for an evaluation of their quality.

**Learning objectives of the subject**

1. Understand how to build a program and use tools the necessary tools: console, editor and compiler.
2. Understand the syntax and semantics of basic expressions and instructions in an imperative programming language (C++).
3. Use functions and actions to develop programs.
4. Understand the concepts of function, action and parameter passing.
5. Understand tables and identify problems for which their use is appropriate.
6. Compare solutions regarding time and memory use and choose the most appropriate solutions for simple cases.
7. Understand search and traversal diagrams.
8. Associate a problem with an appropriate solution scheme.
10. Understand binary search, insertion, sorting, selection, mergesort and quicksort algorithms.
11. Understand other fundamental algorithms: Hömer, fast product.
12. Write programs of about one page in length that are readable, efficient and elegant.

**Study load**

<table>
<thead>
<tr>
<th>Total learning time: 187h 30m</th>
<th>Hours large group: 30h</th>
<th>16.00%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours medium group: 0h</td>
<td></td>
<td>0.00%</td>
</tr>
<tr>
<td>Hours small group: 45h</td>
<td></td>
<td>24.00%</td>
</tr>
<tr>
<td>Guided activities: 7h 30m</td>
<td></td>
<td>4.00%</td>
</tr>
<tr>
<td>Self study: 105h</td>
<td></td>
<td>56.00%</td>
</tr>
</tbody>
</table>
Basic programming principles

Degree competences to which the content contributes:

Description:
Introduction to fundamental concepts: algorithm, program, variable, expression, data type, etc. Basic C++ instructions.

Iterative instructions

Degree competences to which the content contributes:

Description:
For and while instructions. Examples.

Traversal and search diagrams

Degree competences to which the content contributes:

Description:
Sequences. Sequential traversal and search.

Actions and functions

Degree competences to which the content contributes:

Description:
Actions and functions. Parameter passing. Visibility levels.

Recursion

Degree competences to which the content contributes:

Description:
Introduction to recursive design.

Tables

Degree competences to which the content contributes:

Description:
One-dimensional tables. Multidimensional tables. Traversals and searches in tables.
# Tuples

**Degree competences to which the content contributes:**

**Description:**
Programming with tuples.

## Basic algorithms I

**Degree competences to which the content contributes:**

**Description:**
Sorting algorithms. Binary search.

## Basic algorithms II

**Degree competences to which the content contributes:**

**Description:**
Other important algorithms: Hörner, fast product, etc.
## Planning of activities

| Topic development: Basic programming principles | Hours: 11h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 3h  
Guided activities: 0h  
Self study: 6h |
|-----------------------------------------------|--------------------------------------------------|
| Description:  
Understand and assimilate the concepts covered in theory classes. Solve the problems set for this topic, available at www.jutge.org.  
Specific objectives:  
1, 2 |
| Topic development: Iterative instructions | Hours: 11h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 3h  
Guided activities: 0h  
Self study: 6h |
| Description:  
Understand and assimilate the concepts covered in theory classes. Solve the problems set for this topic, available at www.jutge.org.  
Specific objectives:  
1, 2 |
| Topic development: Traversal and search schemes | Hours: 11h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 3h  
Guided activities: 0h  
Self study: 6h |
| Description:  
Understand and assimilate the concepts covered in theory classes. Solve the problems set for this topic, available at www.jutge.org.  
Specific objectives:  
2, 7, 8 |
| Topic development: Actions and functions | Hours: 11h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 3h  
Guided activities: 0h  
Self study: 6h |
**270001 - PRO1 - Programming I**

**Description:**
Understand and assimilate the concepts covered in theory classes. Solve the problems set for this topic, available at www.jutge.org.

**Specific objectives:**
2, 3, 4

---

**Topic development: "Recursion"**

**Description:**
Understand and assimilate the concepts covered in theory classes. Solve the problems set for this topic, available at www.jutge.org.

**Specific objectives:**
3, 4, 9

---

**Consolidation: topics 1 to 5**

**Description:**
Understand and assimilate the concepts covered in theory classes. Solve the problems set for the purpose of consolidating the first part of the course at www.jutge.org.

**Specific objectives:**
1, 2, 3, 4, 7, 8, 9

---

**Topic development: Tables**

**Description:**
Understand and assimilate the concepts covered in theory classes. Solve the problems set for this topic, available at www.jutge.org.

**Specific objectives:**
5, 6, 7, 8, 12
| **Topic development: Tuples** | **Hours**: 11h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 3h  
Guided activities: 0h  
Self study: 6h |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong></td>
<td>Understand and assimilate the concepts covered in theory classes. Solve the problems set for this topic, available at <a href="http://www.jutge.org">www.jutge.org</a>.</td>
</tr>
<tr>
<td><strong>Specific objectives:</strong></td>
<td>12</td>
</tr>
</tbody>
</table>

| **Topic development: Basic algorithms I** | **Hours**: 11h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 3h  
Guided activities: 0h  
Self study: 6h |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong></td>
<td>Understand and assimilate the concepts covered in theory classes. Solve the problems set for this topic, available at <a href="http://www.jutge.org">www.jutge.org</a>.</td>
</tr>
<tr>
<td><strong>Specific objectives:</strong></td>
<td>5, 6, 10, 12</td>
</tr>
</tbody>
</table>

| **Topic development: Basic algorithms II** | **Hours**: 11h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 3h  
Guided activities: 0h  
Self study: 6h |
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Description:</strong></td>
<td>Understand and assimilate the concepts covered in theory classes. Solve the problems set for this topic, available at <a href="http://www.jutge.org">www.jutge.org</a>.</td>
</tr>
<tr>
<td><strong>Specific objectives:</strong></td>
<td>6, 11, 12</td>
</tr>
</tbody>
</table>

| **Consolidation** | **Hours**: 53h 48m  
Theory classes: 6h  
Practical classes: 0h  
Laboratory classes: 9h  
Guided activities: 5h 48m  
Self study: 33h |
| --- | --- |
### Description:
Solve the problems set for this topic, available at www.jutge.org.

**Specific objectives:**
12

### Test 1

<table>
<thead>
<tr>
<th>Description:</th>
<th>Hours: 0h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete, using <a href="http://www.jutge.org">www.jutge.org</a>, a programming exercise. Before being allowed to sit this test, students may be asked to individually resolve a number of exercises from a list.</td>
<td>Guided activities: 0h</td>
</tr>
</tbody>
</table>

**Specific objectives:**
1, 2

<table>
<thead>
<tr>
<th>Hours: 0h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td>Self study: 0h</td>
</tr>
</tbody>
</table>

### Test 2

<table>
<thead>
<tr>
<th>Description:</th>
<th>Hours: 0h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete, using <a href="http://www.jutge.org">www.jutge.org</a>, a programming exercise. Before being allowed to sit this test, students may be asked to individually resolve a number of exercises from a list.</td>
<td>Guided activities: 0h</td>
</tr>
</tbody>
</table>

**Specific objectives:**
3, 4, 7

<table>
<thead>
<tr>
<th>Hours: 0h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td>Self study: 0h</td>
</tr>
</tbody>
</table>

### Test 3

<table>
<thead>
<tr>
<th>Description:</th>
<th>Hours: 0h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete a programming exercises using <a href="http://www.jutge.org">www.jutge.org</a>. Before being allowed to sit this test, students may be asked to individually resolve a number of exercises from a list.</td>
<td>Guided activities: 0h</td>
</tr>
</tbody>
</table>

**Specific objectives:**
5, 6, 8, 9

<table>
<thead>
<tr>
<th>Hours: 0h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td>Self study: 0h</td>
</tr>
</tbody>
</table>

### Final exam

<table>
<thead>
<tr>
<th>Description:</th>
<th>Hours: 0h</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-computer exam. Global course evaluation, emphasizing last chapters</td>
<td>Guided activities: 0h</td>
</tr>
</tbody>
</table>

**Specific objectives:**
10, 11, 12

<table>
<thead>
<tr>
<th>Hours: 0h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td>Self study: 0h</td>
</tr>
</tbody>
</table>
In this course three partial exams C1, C2 and C3 are taken, resulting in grades n1, n2, n3, all ranging from 0 to 10.

With the combined grades of the three exams, a continuous assessment grade AC is computed using formula [1].

\[ AC = 0.25*n1 + 0.3*n2 + 0.45*n3 \]

Students with AC\(\geq\)5 are considered to pass the course with final grade AC.

Students with AC<5 must take the final exam, which results in a grade F, from 0 to 10. The course final grade NF will then be computed with formula [2].

\[ NF = \max (F, (AC + F) / 2) \]

Students with AC\(\geq\)5 may take the final exam if they explicitly request the change to evaluation formula [2]. Procedure to request the change will be announced on the Raco. The change will be irreversible, even if the student changes his/her mind.

A student will have a NP (not shown) if he/she did not take any exam. In another case, the appropriate formula of those above applies.

To be eligible for one exam, it will be necessary to have solved, according to an automatic evaluator (the "Judge"), a minimum of problems from lists associated with each exam, according to the procedure and time limits specified in each case at the "Racó".

Each exam grade is a weighted average of the grades of the exercises that are part of the exam. Exercise grades are, in turn, a weighted average of automatic scores yield by the Judge and manual scores. The weight of automatic or manual scores, counted globally among all exams, will not be less than 30%.

The final exam will be set at the end of the testing period, to distance it in time from C3 and give students time to settle the subject. The final exam will consist of the resolution of several exercises, and correction is entirely manual. Available resources allowing, this examination shall be made on computer and delivered electronically.

The evaluation of the general competence "Autonomous Learning" is based on 2 collected data related to student performance regarding the lists of problems to deliver during the quarter:

Self-learning effort (E):
Average ratio submitted_problem / minimum_required_problems for each exam in the course. It shall be calculated as: 
\[ E = (E1 + E2 + E3) / 3 \]
The metric is saturated in 2. This measure aims to encourage students to do more exercises than the minimum required for each exam.

Learning planning (P):
Measure distribution in time of deliveries of the problems in required lists. It is calculated as 
\[ P = (P1 + P2 + \ldots + Pn) / n \]
Where 
\[ P1 = Si \times \log (Si) + (1-Si) \times \log (1-Si), \]
where Si is the percentage of problems presented in the first half of the delivery period of the list (and therefore 1-Si is the percentage of problems presented in the second half of the period). P is the entropy of the distribution of work between the two halves of the period. A low P value indicates tendency to concentrate effort in one week of expected period of work. A high value indicates a more uniform distribution of the effort.
This metric aims to encourage students to organize work and distribute exercises in time, following the pace of theory and laboratory sessions.

The grade for the general competence will be:
If E \(\leq\) 0.5, the note will be NP (Not enough exercises presented to evaluate competence)
If 0.5 < E < 1 the note is D (Minimum required not reached, indicating little self-learning effort)
If $E \geq 1$, the product $P \times E$ is calculated (ranging from 0 to 2) 
If $P \times E \leq 0.4$ note is C 
If $0.4 < P \times E \leq 1$ note is B 
If $P \times E > 1$, the note will be A

**Bibliography**

**Basic:**

Professorat de Programació 1. Transparències de teoria de l'assignatura (Pàgina web de l'assignatura, apartat material docent).

Professorat de Programació 1. Dotze algorismes fonamentals (Pàgina web de l'assignatura, apartat material docent).

**Complementary:**


**Others resources:**

Hyperlink

http://wwwlsi.upc.edu/~pro1/

https://www.jutge.org

http://www-cs-faculty.stanford.edu/~eroberts/books/ArtAndScienceOfJava/

http://minidosis.org/

http://www.cprogramming.com/