270001 - PRO1 - Programming I

Coordinating unit: 270 - FIB - Barcelona School of Informatics
Teaching unit: 723 - CS - Department of Computer Science
Academic year: 2018
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:
- Edelmira Pasarella Sanchez (edelmira@cs.upc.edu)
- Emma Rollón Rico (erollon@cs.upc.edu)
- Gerard Canal Camprodon (gcanal@iri.upc.edu)
- Jorge Castro Rabal (castro@cs.upc.edu)
- Jose Miguel Rivero Almeida (rivero@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 program's 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 technologies, and versatility to adapt oneself to new situations.
270001 - PRO1 - Programming I

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: Hörner, 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></td>
<td>Hours medium group: 0h</td>
<td>0.00%</td>
</tr>
<tr>
<td></td>
<td>Hours small group: 45h</td>
<td>24.00%</td>
</tr>
<tr>
<td></td>
<td>Guided activities: 7h 30m</td>
<td>4.00%</td>
</tr>
<tr>
<td></td>
<td>Self study: 105h</td>
<td>56.00%</td>
</tr>
<tr>
<td>Content</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------------------------------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Basic programming principles</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Degree competences to which the content contributes:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Introduction to fundamental concepts:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>algorithm, program, variable, expression,</td>
<td></td>
<td></td>
</tr>
<tr>
<td>data type, etc. Basic C++</td>
<td></td>
<td></td>
</tr>
<tr>
<td>instructions.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Iterative instructions</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Degree competences to which the content contributes:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>For and while instructions. Examples.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Traversal and search diagrams</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Degree competences to which the content contributes:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequences. Sequential traversal and search.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Actions and functions</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Degree competences to which the content contributes:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Actions and functions. Parameter passing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Visibility levels.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Recursion</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Degree competences to which the content contributes:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Introduction to recursive design.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Tables</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Degree competences to which the content contributes:</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>One-dimensional tables. Multidimensional</td>
<td></td>
<td></td>
</tr>
<tr>
<td>tables. Traversals and searches in tables.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 270001 - PRO1 - Programming I

<table>
<thead>
<tr>
<th><strong>Tuples</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Degree competences to which the content contributes:</strong></td>
</tr>
<tr>
<td><strong>Description:</strong></td>
</tr>
<tr>
<td>Programming with tuples.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Basic algorithms I</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Degree competences to which the content contributes:</strong></td>
</tr>
<tr>
<td><strong>Description:</strong></td>
</tr>
<tr>
<td>Sorting algorithms. Binary search.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Basic algorithms II</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Degree competences to which the content contributes:</strong></td>
</tr>
<tr>
<td><strong>Description:</strong></td>
</tr>
<tr>
<td>Other important algorithms: Hörner, fast product, etc.</td>
</tr>
</tbody>
</table>
# Planning of activities

## Topic development: Basic programming principles

**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

**Hours:** 11h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 3h  
Guided activities: 0h  
Self study: 6h

## Topic development: Iterative instructions

**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

**Hours:** 11h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 3h  
Guided activities: 0h  
Self study: 6h

## Topic development: Traversal and search schemes

**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

**Hours:** 11h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 3h  
Guided activities: 0h  
Self study: 6h

## Topic development: Actions and functions.

**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, 3, 4

### Topic development: "Recursion"

<table>
<thead>
<tr>
<th>Hours</th>
<th>Description</th>
<th>Specific objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>11h 48m</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>
<td>3, 4, 9</td>
</tr>
</tbody>
</table>

### Consolidation: topics 1 to 5

<table>
<thead>
<tr>
<th>Hours</th>
<th>Description</th>
<th>Specific objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>22h</td>
<td>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 <a href="http://www.jutge.org">www.jutge.org</a>.</td>
<td>1, 2, 3, 4, 7, 8, 9</td>
</tr>
</tbody>
</table>

### Topic development: Tables

<table>
<thead>
<tr>
<th>Hours</th>
<th>Description</th>
<th>Specific objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td>22h 54m</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>
<td>5, 6, 7, 8, 12</td>
</tr>
</tbody>
</table>
## 270001 - PRO1 - Programming I

### Topic development: Tuples

<table>
<thead>
<tr>
<th>Hours</th>
<th>Theory classes: 2h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Practical classes: 0h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 3h</td>
</tr>
<tr>
<td></td>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td></td>
<td>Self study: 6h</td>
</tr>
</tbody>
</table>

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

**Specific objectives:**
12

### Topic development: Basic algorithms I

<table>
<thead>
<tr>
<th>Hours</th>
<th>Theory classes: 2h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Practical classes: 0h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 3h</td>
</tr>
<tr>
<td></td>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td></td>
<td>Self study: 6h</td>
</tr>
</tbody>
</table>

**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, 10, 12

### Topic development: Basic algorithms II

<table>
<thead>
<tr>
<th>Hours</th>
<th>Theory classes: 2h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Practical classes: 0h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 3h</td>
</tr>
<tr>
<td></td>
<td>Guided activities: 0h</td>
</tr>
<tr>
<td></td>
<td>Self study: 6h</td>
</tr>
</tbody>
</table>

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

**Specific objectives:**
6, 11, 12

### Consolidation

<table>
<thead>
<tr>
<th>Hours</th>
<th>Theory classes: 6h</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Practical classes: 0h</td>
</tr>
<tr>
<td></td>
<td>Laboratory classes: 9h</td>
</tr>
<tr>
<td></td>
<td>Guided activities: 5h 48m</td>
</tr>
<tr>
<td></td>
<td>Self study: 33h</td>
</tr>
</tbody>
</table>

**Description:**
Understand and assimilate the concepts covered in theory classes. Solve the problems set for this topic, available at www.jutge.org.
### Test 1

**Description:**
Complete, using www.jutge.org, a programming exercise. Before being allowed to sit this test, students may be asked to individually resolve a number of exercises from a list.

**Specific objectives:**
1, 2

**Hours:** 0h  
Guided activities: 0h  
Self study: 0h

### Test 2

**Description:**
Complete, using www.jutge.org, a programming exercise. Before being allowed to sit this test, students may be asked to individually resolve a number of exercises from a list.

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

**Hours:** 0h  
Guided activities: 0h  
Self study: 0h

### Test 3

**Description:**
Complete a programming exercises using www.jutge.org. Before being allowed to sit this test, students may be asked to individually resolve a number of exercises from a list.

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

**Hours:** 0h  
Guided activities: 0h  
Self study: 0h

### Final exam

**Description:**
On-computer exam. Global course evaluation, emphasizing last chapters

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

**Hours:** 0h  
Guided activities: 0h  
Self study: 0h
In this course three partial exams C1, C2 and C3 are taken. With the combined grades of the three exams, a continuous assessment grade AC is computed using formula [1].

\[ AC = 0.25 \times C1 + 0.3 \times C2 + 0.45 \times C3 \]

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

Students with AC<5 must take the final exam (F). 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 only if they explicitly request the change to evaluation formula [2]. The change will be irreversible.

To be eligible for one exam, it will be necessary to have solved a minimum of problems from lists associated with each exam, according to the procedure and time limits specified in each case at the course web page.

Each exam grade combines an 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%.

Correction of the final exam will be entirely manual.

RE-EVALUATION
Re-evaluation consists of an presential intensive course of 12 hours plus an evaluation, taking place after final exams and before the start of the next semester. Re-evaluation is estimated to require about 50 hours of effort, including sessions, homework, and evaluation.

Minimum requirements to be eligible for re-evaluation:
- Being enrolled in the course
- Having obtained a grade NF between 3.5 and 4.9
- Having taken all course exams (partial and final) during the semester

Requirements to be re-evaluated
- Attend all sessions of the intensive course
- Do the homework and other activities requested by course professors.

Evaluation:
The result of the intensive course evaluation will be "pass" or "fail". The final grade for the course will be:
Final score = 5 if the intensive course score is "pass"
Final score = NF if the intensive course score is "fail"
(where NF is the grade obtained in the regular course evaluation)

GENERAL COMPETENCE

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 $P = (P_1 + P_2 + \ldots + P_n) / n$

Where:
- $P_i = -X_i \log(X_i) - (1-X_i) \log(1-X_i)$, if $X_i < 0.5$
- $P_i = 1$, if $X_i \geq 0.5$

where $X_i$ is the percentage of problems presented in the first half of the delivery period of the list (and therefore $1-X_i$ is the percentage of problems presented in the second half of the period). A low $P$ value indicates tendency to delay effort until the second half of the expected period. A high value indicates a more uniform distribution of the effort, or concentration of the effort in the first half of the period.

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:
- NP if $E \leq 0.5$ (Not enough exercises presented to evaluate competence)
- D if $0.5 < E < 1$ (Minimum required not reached, indicating little self-learning effort)
- C if $E \geq 1$ and $P \cdot E \leq 0.4$
- B if $E \geq 1$ and $0.4 < P \cdot E \leq 1$
- A if $E \geq 1$ and $P \cdot E > 1$
270001 - PRO1 - Programming I

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://www.lsi.upc.edu/~pro1/
  
  https://www.jutge.org
  
  http://www.cs-faculty.stanford.edu/~eroberts/books/ArtAndScienceOfJava/ ava/

  http://minidosis.org/

  http://www.cprogramming.com/