270017 - PROP - Programming Projects

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: 6

Teaching languages: Catalan, Spanish

Teaching staff

Coordinator:
- Alicia Maria Ageno Pulido (ageno@cs.upc.edu)

Others:
- Borja Valles Fuente (valles@cs.upc.edu)
- Elvira Pino Blanco (pino@cs.upc.edu)
- Horacio Rodríguez Hontoria (horacio@cs.upc.edu)
- Jordi Turmo Borràs (turmo@cs.upc.edu)
- Jose Miguel Rivero Almeida (rivero@cs.upc.edu)
- Miquel Sanchez Marre (miquel@cs.upc.edu)

Prior skills

Students are expected to be able to:
- Solve medium-difficulty algorithmic problems from clear specifications and implement their solutions in an imperative programming language.
- Understand basic program structuring mechanisms (modularisation, encapsulation, abstract data types, classes) and apply them to small and medium-sized tasks (a small number of modules).
- Understand object-oriented programming elements (classes, objects, execution mechanisms).
- Use an imperative object-oriented language.
- Use and program data structured in this language.
- Use libraries in this language.
- Master basic error location and correction strategies for simple modules.

Requirements

- Pre-Corequisite IDI
- Pre-Corequisite IES
- Prerequisite EDA

Degree competences to which the subject contributes

Specific:
CT2.3. To design, develop, select and evaluate computer applications, systems and services and, at the same time, ensure its reliability, security and quality in function of ethical principles and the current legislation and normative.
CT2.4. To demonstrate knowledge and capacity to apply the needed tools for storage, processing and access to the information system, even if they are web-based systems.
CT2.5. To design and evaluate person-computer interfaces which guarantee the accessibility and usability of computer systems, services and applications.
CT3.6. To demonstrate knowledge about the ethical dimension of the company: in general, the social and corporative responsibility and, concretely, the civil and professional responsibilities of the informatics engineer.

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.

CT4.3. To demonstrate knowledge and capacity to apply the fundamental principles and the basic techniques of the intelligent systems and its practical application.

CT5.1. To choose, combine and exploit different programming paradigms, at the moment of building software, taking into account criteria like ease of development, efficiency, portability and maintainability.

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.

CT5.5. To use the tools of a software development environment to create and develop applications.

CT5.6. To demonstrate knowledge and capacity to apply the fundamental principles and basic techniques of parallel, concurrent, distributed and real-time programming.

CT6.1. To demonstrate knowledge and capacity to manage and maintain computer systems, services and applications.

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.

CT8.7. To control project versions and configurations.

Generical:

G1. ENTREPRENEURSHIP AND INNOVATION: to know and understand the organization of a company and the sciences which govern its activity; capacity to understand the labour rules and the relation between planning, industrial and business strategies, quality and benefit. To develop creativity, entrepreneur spirit and innovation tendency.

G5. TEAMWORK: to be capable to work as a team member, being just one more member or performing management tasks, with the finality of contributing to develop projects in a pragmatic way and with responsibility sense; to assume compromises taking into account the available resources.
270017 - PROP - Programming Projects

Teaching methodology

Topics will be explained in a practical way through the use of numerous examples.

The theory lessons will introduce the necessary knowledge, techniques and concepts for the course project. Some of the laboratory classes (probably towards the beginning of the course) will be used to briefly present notations, languages, libraries and tools. In most laboratory classes, however, students take the initiative. A number of hours are available for project groups to work together and to discuss any doubts they have with the lecturer.

The two-hour theory classes and two-hour laboratory classes will take place once a week, except for the last two weeks (laboratory classes only). First week there are no laboratory classes.

The bulk of work on the subject will be on the project, which will have a strong algorithmic and data structure component.

The approach to implementing this project will be as follows:
- Students (12-15) will form five groups of three, called a cluster.
- Each cluster will be assigned a tutor.
- Within each group, a single student will be responsible for class programming and testing.

The programming language used will be Java.

Learning objectives of the subject

1. Use previously acquired programming skills to develop a medium-sized programming project, select appropriate algorithms and data structures and build a program correctly and efficiently.
2. Organize the design and implementation work of a medium-sized team (three to four people) according to a predetermined schedule. This organization includes both the overall planning and the allocation of tasks between group members.
3. Identify program specification, design and implementation components that can be factorised and immediately resolved and use object-orientation (OO) mechanisms for effective factorisation.
4. Understand object-oriented design and programming principles and the advantages and disadvantages of adopting this programming paradigm.
5. Use class and inheritance mechanisms to avoid redundancy and capture specification, design and implementation abstractions. They should also understand the advantages of this use of object orientation in terms of reliability, modifiability, portability, reusability and productivity.
6. Deal effectively with anomalous program situations using, if necessary, the exception mechanism provided by the programming language.
7. Design a reasonably usable and effective user interface for a program using a GUI library.
8. Develop (in groups) a medium-sized programming project, according to a pre-defined development plan and architecture.
# Study Load

<table>
<thead>
<tr>
<th>Component</th>
<th>Hours</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory classes</td>
<td>30h</td>
<td>20.00%</td>
</tr>
<tr>
<td>Practical classes</td>
<td>0h</td>
<td>0.00%</td>
</tr>
<tr>
<td>Laboratory classes</td>
<td>30h</td>
<td>20.00%</td>
</tr>
<tr>
<td>Guided activities</td>
<td>6h</td>
<td>4.00%</td>
</tr>
<tr>
<td>Self study</td>
<td>84h</td>
<td>56.00%</td>
</tr>
</tbody>
</table>

**Total learning time:** 150h
# Content

## ISE review: Specification, class diagrams and use cases.

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

**Description:**

## Introduction to Java and OO concepts in Java.

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

**Description:**
Object-oriented (OO) specification, design and programming. Classes, objects, attributes, methods and relationships. Encapsulation and hiding, abstraction and classification, inheritance. Polymorphism, static and dynamic binding. Other relationships: association, aggregation, dependency.

## Design and implementation in Java of the Three-tier architecture

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

**Description:**
The three-tier architecture design paradigm and its Java implementation

## Program debugging

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

**Description:**

## Design patterns in Java

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

**Description:**
Some of the classical design patterns will be explained and implemented in Java (decorator, state, singleton, etc)

## Basic interface design concepts

**Degree competences to which the content contributes:**
**Description:**
Input and output types. Principles of communication with humans. Information presentation, data entry and validation, navigation, support levels and user models.

**Software development toolchain**

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

**Description:**
There will be an overview of the typical toolchain to develop software: version control (git), testing integration (junit)
## Planning of activities

<table>
<thead>
<tr>
<th>Activity</th>
<th>Hours</th>
<th>Description</th>
<th>Specific objectives</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Project description and group creation</strong></td>
<td><strong>3h</strong></td>
<td><strong>Description:</strong> Description of project deliveries. Group creation for practical sessions.</td>
<td>2, 8</td>
</tr>
<tr>
<td><strong>Introduction to Java and Basic OO concepts in Java</strong></td>
<td><strong>16h</strong></td>
<td><strong>Description:</strong> Topic 2 development.</td>
<td>4, 5, 8</td>
</tr>
<tr>
<td><strong>Design and implementation in Java of the Three-tier architecture</strong></td>
<td><strong>6h</strong></td>
<td><strong>Description:</strong> Topic 3 development.</td>
<td>3, 4, 5, 8</td>
</tr>
<tr>
<td><strong>Program debugging</strong></td>
<td><strong>10h</strong></td>
<td><strong>Description:</strong></td>
<td></td>
</tr>
</tbody>
</table>
## Design patterns in Java

**Description:**
Some classic design patterns will be introduced (or reviewed if already known from the ISE course) and implemented in Java.

**Specific objectives:**
4, 5

<table>
<thead>
<tr>
<th>Hours</th>
<th>Theory classes: 4h</th>
<th>Practical classes: 0h</th>
<th>Laboratory classes: 0h</th>
<th>Guided activities: 0h</th>
<th>Self study: 4h</th>
</tr>
</thead>
</table>

## Java: implementing interfaces

**Description:**
Topic 6 development.

**Specific objectives:**
7, 8

<table>
<thead>
<tr>
<th>Hours</th>
<th>Theory classes: 4h</th>
<th>Practical classes: 0h</th>
<th>Laboratory classes: 0h</th>
<th>Guided activities: 0h</th>
<th>Self study: 4h</th>
</tr>
</thead>
</table>

## First delivery of the project

**Description:**
Implementation principles and design. Students will fully implement the domain model and the basic algorithmic kernel for the project, provide test suites for the implemented part and provide a preliminary design of the displays configuring the application interface.

**Specific objectives:**
1, 2, 3, 4, 5, 6, 8

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

---

**Description:**

Task 4 development.

**Specific objectives:**
6, 8
### Theory Test

**Hours:** 17h  
- Guided activities: 2h  
- Self study: 15h

**Description:**  
This exam will assess the theory content.

**Specific objectives:**  
1, 4, 5

### Second delivery of the project

**Hours:** 2h  
- Guided activities: 2h  
- Self study: 0h

**Description:**  
Final project delivery. The documentation should include the diagram of classes (UML), full implementation of the project and the test suites as appropriate, and the user manual. Each group will also give a presentation of their project.

**Specific objectives:**  
1, 2, 3, 4, 5, 6, 7, 8

### Supervision of first assignment

**Hours:** 30h  
- Theory classes: 0h  
- Practical classes: 0h  
- Laboratory classes: 10h  
- Guided activities: 0h  
- Self study: 20h

**Description:**  
Laboratory classes related to the first assignment.

**Specific objectives:**  
1, 2, 3, 4, 5, 6, 8

### Supervision of second assignment

**Hours:** 40h  
- Theory classes: 0h  
- Practical classes: 0h  
- Laboratory classes: 12h  
- Guided activities: 0h  
- Self study: 28h

**Description:**  
Laboratory classes related to the second assignment.

**Specific objectives:**  
1, 2, 3, 4, 5, 6, 7, 8
Second assignment presentation

| Hours: 7h 30m |
| Theory classes: 0h |
| Practical classes: 0h |
| Laboratory classes: 0h |
| Guided activities: 7h 30m |
| Self study: 0h |

**Description:**
Presentation of the second assignment and interview.

**Specific objectives:**
8

Qualification system

Final mark: 0.20 * Theory_Exam_Mark + 0.80 * Project_Mark

Project mark: (0.40 * Delivery1 + 0.60 * Delivery2) * FT

Given that the project is team work in which all group members (and only group members) must participate, the final project mark will be weighted according to a work factor (FT). This work factor is a score (0 <= FT <= 1) representing the work of each group member. The work factor will be determined on the basis of task distribution within the group (first and second delivery) and the final presentation of the group. Work factors will be published together with the second project delivery marks.

Transversal competency in teamwork: Assessment will be based on a simple formula, with the group tutor scoring competency aspects for each group member.

Transversal competency in entrepreneurship and innovation: Assessment will be based on a simple formula, with the group tutor scoring competency aspects for each group member.
Bibliography

Basic:


Complementary:


Others resources:

Hyperlink

https://www.oracle.com/java/index.html

http://argouml.tigris.org/