Course guides
340380 - PROP-I4023 - Programming Project

Unit in charge: Vilanova i la Geltrú School of Engineering
Teaching unit: 723 - CS - Department of Computer Science.
Degree: BACHELOR'S DEGREE IN INFORMATICS ENGINEERING (Syllabus 2018). (Compulsory subject).
Academic year: 2021 ECTS Credits: 6.0 Languages: Catalan

LECTURER
Coordinating lecturer: Orellana Bech, Bernat
Others: Orellana Bech, Bernat

PRIOR SKILLS
Knowledge of programming and data structures:

- Ability to solve algorithmic problems of medium difficulty from a clear specification, and implement solutions in an imperative programming language.

- Knowledge of basic mechanisms for structuring programs (modularization, encapsulation, abstract data types, classes) and ability to apply them to problems small-sized (a few modules)

- Knowledge of the elements of object oriented programming (classes, objects, mechanisms for implementation).

- Familiarity with object-oriented imperative language.

- Ability to use data structures and programming in this language.

- Ability to use language in this book.

- Mastery of basic strategies for finding and correcting errors in simple modules.

DEGREE COMPETENCES TO WHICH THE SUBJECT CONTRIBUTES
Specific:
1. CECO1. Ability to have a thorough understanding of the fundamental principles and models of computation, ability to apply the principles to interpret, select, evaluate, model, and create new concepts, theories, applications and advance the technological development related to computing.
2. CECO2. Ability to understand theoretical basics of programming languages and techniques of lexical, syntactic and semantic associates processing, and apply them to create, design and process languages.
3. CECO3. Ability to assess the computational complexity of a problem, to know algorithmic strategies that may lead to its resolution and to recommend, develop and implement the one which guarantees the best performance according to established requirements.
4. CECO4. Ability to learn basics, paradigms and techniques of intelligent systems and analyze, design and build systems, services and computing applications that use these techniques in any scope.

TEACHING METHODOLOGY
In the course we work is algorithmic programming techniques through lectures and laboratory classes. In the laboratory classes we look at object oriented programming in a practice, developing programming activities to establish these techniques and finally developing a project of average size for which students must develop the techniques learned in lectures and combine them with object-oriented programming techniques that have been in the laboratory classes.
LEARNING OBJECTIVES OF THE SUBJECT

Learning techniques to identify the complexity of a problem and apply the appropriate resolution strategy.
Estrucutra learning of graph to represent combinatorial problems.
Learning the different algorithmic strategies for solving computational problems.
Learn advanced concepts of OOP.

STUDY LOAD

<table>
<thead>
<tr>
<th>Type</th>
<th>Hours</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hours small group</td>
<td>30.0</td>
<td>20.00</td>
</tr>
<tr>
<td>Self study</td>
<td>90.0</td>
<td>60.00</td>
</tr>
<tr>
<td>Hours large group</td>
<td>30.0</td>
<td>20.00</td>
</tr>
</tbody>
</table>

Total learning time: 150 h

CONTENTS

**Graphs**

**Description:**
Review of graph’s theory. Graph representation for computing: adjacency matrix, adjacency lists. Classical graph algorithms. graph traversal.

**Specific objectives:**
(ENG) Aprenentatge de la importància dels grafs per representar problemes on hi ha relacions entre elements. Aprenentatge de diferents estructures de representació pels grafs i la eficiència espacial i temporal de les mateixes Disseny d'algorisme de connectivitat entre nodes del graf: Camins, parts connexes d'un graf, cicles, etc.

**Full-or-part-time:** 1h
Theory classes: 1h

**Algorithmic costs**

**Description:**

**Specific objectives:**
(ENG) Adquirir i el concepte de cost temporal i espaiàl d'un problema. Aprenentatge de la jerarquia de complexitat. Capacitat de discernir un problema tractable d'un d'intractable. Aprenentage de la tècnica de reducció.

**Full-or-part-time:** 3h
Theory classes: 3h
<table>
<thead>
<tr>
<th>Course</th>
<th>Description</th>
<th>Full-or-part-time</th>
<th>Theory classes</th>
<th>Laboratory classes</th>
<th>Self study</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Combinatorial Algorithms</strong></td>
<td>Application of generative algorithms. Search for any solution or for the optimal solution. Description of the search space. Generative search scheme. Blind search. Heuristic functions. Informed search. Game search algorithms: Minimax, alpha-beta, Monte Carlo.</td>
<td>26h</td>
<td>14h</td>
<td>4h</td>
<td>8h</td>
</tr>
<tr>
<td><strong>Greedy algorithms</strong></td>
<td>Scheme of the greedy algorithms. Greedy criterion: the need for optimality demonstration. Example cases.</td>
<td>12h</td>
<td>4h</td>
<td></td>
<td>8h</td>
</tr>
<tr>
<td><strong>Dynamic Programming</strong></td>
<td>Inefficient recursive programs. Memorization. Steps of dynamic programming. Recursive equations.</td>
<td>6h</td>
<td>6h</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Agregations and divide and conquer</strong></td>
<td>Divide and conquer scheme. Aggregation and elimination scheme.</td>
<td>2h</td>
<td>2h</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Object Oriented Language: Java</strong></td>
<td>Hands-on learning of the Java language</td>
<td>5h</td>
<td>5h</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Advanced Object Oriented Programming

Description:
Learning advanced features of object-oriented programming.

Specific objectives:
Learning advanced features of object-oriented programming.

Full-or-part-time: 3h
Laboratory classes: 3h

ACTIVITIES

Activity 1

Description:
Working with preprogrammed classes and their extension. Use of Javadoc. Use of inheritance and polymorphism.

Full-or-part-time: 21h
Laboratory classes: 5h
Self study: 16h

Activity 2

Description:
Object-Oriented Programming of combinatorial algorithms for games.

Full-or-part-time: 23h
Laboratory classes: 5h
Self study: 16h

Project

Description:
Programming of a complex optimization problem involving dynamic and/or combinatorial and/or greedy programming algorithms. Use of UML modeling tools and Javadoc. Use of Git repository for team working.

Full-or-part-time: 48h 40m
Laboratory classes: 8h 40m
Self study: 40h

GRADING SYSTEM

Theory Grade = max(0,5 Exam1 + 0,5 Exam2; second-chance examination)
Small projects grade = 0,5 Small project 1 + 0,5 small project 2

IF Theory Grade >= 3 then Final Grade = 0,5 Theory + 0,3 Big project + 0,2 Small projects
Else Final Grade = 0,7 Theory + 0,2 Big project + 0,1 Small projects
BIBLIOGRAPHY

Basic: