270025 - LP - Programming Languages

Coordinating unit: 270 - FIB - Barcelona School of Informatics
Teaching unit: 723 - CS - Department of Computer Science
Academic year: 2019
Degree: BACHELOR'S DEGREE IN INFORMATICS ENGINEERING (Syllabus 2010). (Teaching unit Optional)
ECTS credits: 6
Teaching languages: Catalan

Teaching staff

Coordinator: - Jordi Petit Silvestre (jpetit@cs.upc.edu)
Others: - Edelmira Pasarella Sanchez (edelmira@cs.upc.edu)
- Fernando Orejas Valdés (orejas@cs.upc.edu)
- Gerard Escudero Bakx (gerard.escudero@upc.edu)
- Jose Carmona Vargas (jcarmona@cs.upc.edu)

Prior skills

Prior skills on logic acquired in the course on Foundations of Mathematics (FM):
- Knowledge of basic concepts of logic propositions and predicates
- Knowledge of logical inference.

Prior programming skills acquired in the courses on Data structures and algorithms (EDA) and Programming Projects (PROP):
- Knowledge of programming.
- Knowledge of object oriented programming languages

Requirements

- Corequisite PROP

Degree competences to which the subject contributes

Specific:
CCO1.2. To demonstrate knowledge about the theoretical fundamentals of programming languages and the associated lexical, syntactical and semantic processing techniques and be able to apply them to create, design and process languages.

General:
G4. EFFECTIVE ORAL AND WRITTEN communication: To communicate with other people knowledge, procedures, results and ideas orally and in a written way. To participate in discussions about topics related to the activity of a technical informatics engineer.
G6. SOLVENT USE OF THE INFORMATION RESOURCES: To manage the acquisition, structuring, analysis and visualization of data and information of the field of the informatics engineering, and value in a critical way the results of this management.
270025 - LP - Programming Languages

Teaching methodology

Classes are divided into theory and lab sessions.

In the theory sessions the concepts of the subject will be developed. The presentation of new theoretical material is combined with the resolution of examples and the interaction with the students to promote the discussion of introduced concepts.

In the laboratory classes the introduced concepts will be put into practice and applied to specific problems and programming languages. There will be three small practices associated with the different evaluation tests.

Learning objectives of the subject

1. Knowing the different compilation steps, including lexical, syntactic and semantic.
2. Conocer herramientas para la creación de analizadores léxicos y sintácticos.
3. Knowing the differences between a compiler and an interpreter.
4. Distinguish the main features of a given programming language. Identify its strengths and weaknesses and be able to justify a choice.
5. Understand the main features of functional programming languages.
6. Learn advanced programming language concepts.
7. Being able to model and specify hardware or software problems using functional languages.
8. Learn the main features of scripting languages.
9. Ability to learn new programming languages independently.

Study load

<table>
<thead>
<tr>
<th>Total learning time: 150h</th>
<th>Hours large group: 30h</th>
<th>20.00%</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: 30h</td>
<td>20.00%</td>
</tr>
<tr>
<td></td>
<td>Guided activities: 6h</td>
<td>4.00%</td>
</tr>
<tr>
<td></td>
<td>Self study: 84h</td>
<td>56.00%</td>
</tr>
</tbody>
</table>
### Content

<table>
<thead>
<tr>
<th>Topic</th>
<th>Degree competences to which the content contributes:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Introduction to programming languages.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description:</strong></td>
<td>Turing completeness. Programming paradigms. Main features.</td>
</tr>
<tr>
<td><strong>Introduction to compilers.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Functional languages.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description:</strong></td>
<td>Foundations. Pattern Matching. Eager/Lazy evaluation. Introduction to Haskell.</td>
</tr>
<tr>
<td><strong>Type systems.</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description:</strong></td>
<td>Types in programming languages. Types and reliability. Type systems in functional languages and object-oriented languages.</td>
</tr>
<tr>
<td><strong>Higher-order programming</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description:</strong></td>
<td>Higher-order functions. Applications. Programming higher-order functions in functional and object-oriented languages.</td>
</tr>
<tr>
<td><strong>Modeling and Specification using functional languages</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Description:</strong></td>
<td>Modeling and specification using declarative languages. Prototyping. Executable specifications. Program transformation.</td>
</tr>
</tbody>
</table>
### Scripting languages

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

**Description:**
## Planning of activities

<table>
<thead>
<tr>
<th>Course</th>
<th>Hours</th>
<th>Theory classes:</th>
<th>Practical classes:</th>
<th>Laboratory classes:</th>
<th>Guided activities:</th>
<th>Self study:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction to programming languages</td>
<td>6h</td>
<td>2h</td>
<td>0h</td>
<td>0h</td>
<td>0h</td>
<td>4h</td>
</tr>
<tr>
<td>Specific objectives:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4, 5, 9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Introduction to compilers</td>
<td>22h</td>
<td>6h</td>
<td>0h</td>
<td>8h</td>
<td>0h</td>
<td>8h</td>
</tr>
<tr>
<td>Specific objectives:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 2, 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CL</td>
<td>10h</td>
<td></td>
<td></td>
<td></td>
<td>2h</td>
<td>8h</td>
</tr>
<tr>
<td>Specific objectives:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1, 2, 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functional languages</td>
<td>18h</td>
<td>4h</td>
<td>0h</td>
<td>6h</td>
<td>0h</td>
<td>8h</td>
</tr>
<tr>
<td>Specific objectives:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4, 5, 9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Type systems</td>
<td>10h</td>
<td>4h</td>
<td>0h</td>
<td>0h</td>
<td>0h</td>
<td>6h</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Specific objectives:
4, 5, 6, 9

| Higher-order programming | Hours: 10h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 2h  
Guided activities: 0h  
Self study: 6h |
|--------------------------|-----------------|
| Modeling and specification using functional languages | Hours: 12h  
Theory classes: 4h  
Practical classes: 0h  
Laboratory classes: 2h  
Guided activities: 0h  
Self study: 6h |
| EP | Hours: 12h  
Guided activities: 2h  
Self study: 10h |
| Scripting languages | Hours: 14h  
Theory classes: 2h  
Practical classes: 0h  
Laboratory classes: 6h  
Guided activities: 0h  
Self study: 6h |
| Specific objectives: | 5, 7 |
| Specific objectives: | 5, 6 |
| Specific objectives: | 3, 4, 8, 9 |
### Learning a new programming language

*Hours:* 10h  
- Theory classes: 2h  
- Practical classes: 0h  
- Laboratory classes: 2h  
- Guided activities: 2h  
- Self study: 4h

### CT

*Hours:* 7h  
- Guided activities: 1h  
- Self study: 6h

**Specific objectives:**  
4, 9

### Reviewing activities

*Hours:* 4h  
- Theory classes: 2h  
- Practical classes: 0h  
- Laboratory classes: 2h  
- Guided activities: 0h  
- Self study: 0h

### EF

*Hours:* 15h  
- Guided activities: 3h  
- Self study: 12h

**Specific objectives:**  
4, 5, 6, 7, 8
Qualification system

The grade is

\[ N = 0.35 \cdot F + 0.30 \cdot P + 0.25 \cdot L + 0.10 \cdot D \]

where:

- \( F \) = grade of the final exam
- \( P \) = grade of the partial exam
- \( L \) = grade of the project
- \( D \) = note of the directed work

The final exam will be a written exam that will evaluate all the contents of the course.

The partial exam will be a computer exam to solve programming problems using Haskell.

The project consists in using tools to generate compilers and Python to solve a practical case.

The directed work consists in preparing a video and a written document about the properties of a programming language. Your assessment will be done by peer evaluation (co-evaluation).

The qualifications of the transversal competences are obtained from the directed work.

Bibliography

Basic: