Course guide
270402 - PA1 - Programming and Algorithms I

Unit in charge: Barcelona School of Informatics
Teaching unit: 723 - CS - Department of Computer Science.

Degree: BACHELOR’S DEGREE IN ARTIFICIAL INTELLIGENCE (Syllabus 2021). (Compulsory subject).

Academic year: 2022 ECTS Credits: 6.0 Languages: Catalan, Spanish

LECTURER
Coordinating lecturer: JORDI DELGADO PIN
Others: Primer quadrimestre:
JOSÉ LUIS BALCÁZAR NAVARRO - 11, 12
JORDI DELGADO PIN - 11, 12

PRIOR SKILLS
Students are expected to have acquired the knowledge and skills defined for the scientific-technical branch of upper secondary school education or equivalent.

DEGREE COMPETENCES TO WHICH THE SUBJECT CONTRIBUTES

Specific:
CE02. To master the basic concepts of discrete mathematics, logic, algorithmic and computational complexity, and its application to the automatic processing of information through computer systems. To be able to apply all these for solving problems.
CE03. To identify and apply the basic algorithmic procedures of computer technologies to design solutions to problems by analyzing the suitability and complexity of the proposed algorithms.
CE04. To design and use efficiently the most appropriate data types and structures to solve a problem.
CE10. To analyze, design, build and maintain applications in a robust, secure and efficient way, choosing the most appropriate paradigm and programming languages.
CE12. To master the fundamental principles and models of computing and to know how to apply them in order to interpret, select, assess, model, and create new concepts, theories, uses and technological developments related to artificial intelligence.
CE13. To evaluate the computational complexity of a problem, identify algorithmic strategies that can lead to its resolution and recommend, develop and implement the one that guarantees the best performance in accordance with the established requirements.

Generical:
CG2. To use the fundamental knowledge and solid work methodologies acquired during the studies to adapt to the new technological scenarios of the future.
CG4. Reasoning, analyzing reality and designing algorithms and formulations that model it. To identify problems and construct valid algorithmic or mathematical solutions, eventually new, integrating the necessary multidisciplinary knowledge, evaluating different alternatives with a critical spirit, justifying the decisions taken, interpreting and synthesizing the results in the context of the application domain and establishing methodological generalizations based on specific applications.
CG8. Perform an ethical exercise of the profession in all its facets, applying ethical criteria in the design of systems, algorithms, experiments, use of data, in accordance with the ethical systems recommended by national and international organizations, with special emphasis on security, robustness, privacy, transparency, traceability, prevention of bias (race, gender, religion, territory, etc.) and respect for human rights.

Transversal:
CT4. Teamwork. Be able to work as a member of an interdisciplinary team, either as a member or conducting management tasks, with the aim of contributing to develop projects with pragmatism and a sense of responsibility, taking commitments taking into account available resources.
CT6. Autonomous Learning. Detect deficiencies in one’s own knowledge and overcome them through critical reflection and the choice of the best action to extend this knowledge.
**TEACHING METHODOLOGY**

Teaching the course is structured in lectures and laboratory sessions.

Teachers will use lectures to introduce the essential contents of the course. In the laboratory sessions the contents of the course will be brought to the computer by carrying out practical problems. The laboratory classes will be a continuation of the lectures, where new concepts will be implemented as they appear in lectures.

**LEARNING OBJECTIVES OF THE SUBJECT**

1. Learn the most basic concepts of programming with functions (simple and high-order) and their use as tools to adapt a programming language to the domain of the problem to be solved
2. Learn the concept of recursion and how to distinguish iterative and recursive processes defined with recursive functions
3. Initiate the student in the understanding of the concept of software design
4. Being able to deal with the idea of an error in a program and using exceptions to cope with these errors
5. Learn the concept of container and the situations in which it is relevant to use them. Know the cost of its use (informally)
6. Introduce the student to the computation of the complexity of algorithms
7. Involve the student in the design and implementation of a simple problem collaborating with other students

**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**

**Building Abstractions with Functions: Functions (simple and high-order). Environments. Execution flow control.**

**Description:**
The student is introduced to the main problem-solving mechanism, from a bottom-up point of view, of designing functions that bring the programming language closer to the domain of the problem. The mechanisms of function definition, parameter passing, environment and scope, and the main control structures of the execution flow (conditional, iterative) are studied.

**Building Abstractions with Functions: Recursion and examples.**

**Description:**
The fact that a function is able to call itself is introduced, and how this leads to recursion as a conceptual mechanism for solving problems, and the idea of process. Recursive functions may define iterative and recursive processes.

**Building Abstractions with Functions: Design. Error handling with exceptions.**

**Description:**
The concept of designing a program to solve a problem is introduced. The idea of error in a program and how to deal with them by means of the exception mechanism is introduced.
**Building Abstractions with Functions: Examples of solving problems**

**Description:**
The concepts introduced so far are reinforced with numerous problems of low and medium difficulty.

**Building Abstractions with Functions: Containers**

**Description:**
We start working with containers provided by the programming language: Sequences, Lists, Dictionaries and others implemented within the course, such as Matrices or Trees.

**Building Abstractions with Functions: Mutability. Iterators and Generators.**

**Description:**
The concept of mutability is first seen in relation to lists, and the advantages and disadvantages of having immutable data structures are discussed. Iterators and generators are introduced as new control structures.

**Introduction to Algorithm Complexity**

**Description:**
A short introduction is made to asymptotic notation and to the complexity in the worst case of some notable algorithms seen during the course.
ACTIVITIES

**Building Abstractions with Functions: Functions and Control Structures**

**Description:**
The student should pay attention to the lecture and he/she should work through the exercises suggested by the lecturer.

**Specific objectives:**
1, 2, 3, 4

**Related competencies:**
CG4. Reasoning, analyzing reality and designing algorithms and formulations that model it. To identify problems and construct valid algorithmic or mathematical solutions, eventually new, integrating the necessary multidisciplinary knowledge, evaluating different alternatives with a critical spirit, justifying the decisions taken, interpreting and synthesizing the results in the context of the application domain and establishing methodological generalizations based on specific applications.
CG2. To use the fundamental knowledge and solid work methodologies acquired during the studies to adapt to the new technological scenarios of the future.
CE10. To analyze, design, build and maintain applications in a robust, secure and efficient way, choosing the most appropriate paradigm and programming languages.
CE03. To identify and apply the basic algorithmic procedures of computer technologies to design solutions to problems by analyzing the suitability and complexity of the proposed algorithms.
CE12. To master the fundamental principles and models of computing and to know how to apply them in order to interpret, select, assess, model, and create new concepts, theories, uses and technological developments related to artificial intelligence.
CT6. Autonomous Learning. Detect deficiencies in one’s own knowledge and overcome them through critical reflection and the choice of the best action to extend this knowledge.

**Full-or-part-time: 48h**
Theory classes: 12h
Laboratory classes: 12h
Self study: 24h
Building Abstractions with Functions: Containers, Iterators, Generators.

Description:
The student should pay attention to the lecture and he/she should work through the exercises suggested by the lecturer.

Specific objectives:
5, 7

Related competencies:
CG8. Perform an ethical exercise of the profession in all its facets, applying ethical criteria in the design of systems, algorithms, experiments, use of data, in accordance with the ethical systems recommended by national and international organizations, with special emphasis on security, robustness, privacy, transparency, traceability, prevention of bias (race, gender, religion, territory, etc.) and respect for human rights.
CG4. Reasoning, analyzing reality and designing algorithms and formulations that model it. To identify problems and construct valid algorithmic or mathematical solutions, eventually new, integrating the necessary multidisciplinary knowledge, evaluating different alternatives with a critical spirit, justifying the decisions taken, interpreting and synthesizing the results in the context of the application domain and establishing methodological generalizations based on specific applications.
CG2. To use the fundamental knowledge and solid work methodologies acquired during the studies to adapt to the new technological scenarios of the future.
CE02. To master the basic concepts of discrete mathematics, logic, algorithmic and computational complexity, and its application to the automatic processing of information through computer systems. To be able to apply all these for solving problems.
CE10. To analyze, design, build and maintain applications in a robust, secure and efficient way, choosing the most appropriate paradigm and programming languages.
CE03. To identify and apply the basic algorithmic procedures of computer technologies to design solutions to problems by analyzing the suitability and complexity of the proposed algorithms.
CE04. To design and use efficiently the most appropriate data types and structures to solve a problem.
CT6. Autonomous Learning. Detect deficiencies in one’s own knowledge and overcome them through critical reflection and the choice of the best action to extend this knowledge.
CT4. Teamwork. Be able to work as a member of an interdisciplinary team, either as a member or conducting management tasks, with the aim of contributing to develop projects with pragmatism and a sense of responsibility, taking commitments taking into account available resources.

Full-or-part-time: 56h
Theory classes: 14h
Laboratory classes: 14h
Self study: 28h

Algorithm Complexity

Description:
The student should pay attention to the lecture and he/she should work through the exercises suggested by the lecturer.

Specific objectives:
6

Related competencies:
CE13. To evaluate the computational complexity of a problem, identify algorithmic strategies that can lead to its resolution and recommend, develop and implement the one that guarantees the best performance in accordance with the established requirements.

Full-or-part-time: 16h
Theory classes: 4h
Laboratory classes: 4h
Self study: 8h
Midterm Exam

Specific objectives:
1, 2, 3, 4

Related competencies:
CG4. Reasoning, analyzing reality and designing algorithms and formulations that model it. To identify problems and construct valid algorithmic or mathematical solutions, eventually new, integrating the necessary multidisciplinary knowledge, evaluating different alternatives with a critical spirit, justifying the decisions taken, interpreting and synthesizing the results in the context of the application domain and establishing methodological generalizations based on specific applications.

CG2. To use the fundamental knowledge and solid work methodologies acquired during the studies to adapt to the new technological scenarios of the future.

CE10. To analyze, design, build and maintain applications in a robust, secure and efficient way, choosing the most appropriate paradigm and programming languages.

CE03. To identify and apply the basic algorithmic procedures of computer technologies to design solutions to problems by analyzing the suitability and complexity of the proposed algorithms.

CE12. To master the fundamental principles and models of computing and to know how to apply them in order to interpret, select, assess, model, and create new concepts, theories, uses and technological developments related to artificial intelligence.

CT6. Autonomous Learning. Detect deficiencies in one's own knowledge and overcome them through critical reflection and the choice of the best action to extend this knowledge.

Full-or-part-time: 6h
Guided activities: 2h
Self study: 4h
**Final Exam**

**Specific objectives:**
5, 6, 7

**Related competencies:**
CG8. Perform an ethical exercise of the profession in all its facets, applying ethical criteria in the design of systems, algorithms, experiments, use of data, in accordance with the ethical systems recommended by national and international organizations, with special emphasis on security, robustness, privacy, transparency, traceability, prevention of bias (race, gender, religion, territory, etc.) and respect for human rights.

CG4. Reasoning, analyzing reality and designing algorithms and formulations that model it. To identify problems and construct valid algorithmic or mathematical solutions, eventually new, integrating the necessary multidisciplinary knowledge, evaluating different alternatives with a critical spirit, justifying the decisions taken, interpreting and synthesizing the results in the context of the application domain and establishing methodological generalizations based on specific applications.

CG2. To use the fundamental knowledge and solid work methodologies acquired during the studies to adapt to the new technological scenarios of the future.

CE13. To evaluate the computational complexity of a problem, identify algorithmic strategies that can lead to its resolution and recommend, develop and implement the one that guarantees the best performance in accordance with the established requirements.

CE10. To analyze, design, build and maintain applications in a robust, secure and efficient way, choosing the most appropriate paradigm and programming languages.

CE02. To master the basic concepts of discrete mathematics, logic, algorithmic and computational complexity, and its application to the automatic processing of information through computer systems. To be able to apply all these for solving problems.

CE04. To design and use efficiently the most appropriate data types and structures to solve a problem.

CE03. To identify and apply the basic algorithmic procedures of computer technologies to design solutions to problems by analyzing the suitability and complexity of the proposed algorithms.

CT6. Autonomous Learning. Detect deficiencies in one’s own knowledge and overcome them through critical reflection and the choice of the best action to extend this knowledge.

CT4. Teamwork. Be able to work as a member of an interdisciplinary team, either as a member or conducting management tasks, with the aim of contributing to develop projects with pragmatism and a sense of responsibility, taking commitments taking into account available resources.

**Full-or-part-time:** 9h
Guided activities: 3h
Self study: 6h

---

**GRADING SYSTEM**

Grading the course will consist of two theoretical tests (T1 and T2), one mid-course and the other at the end, and one medium-sized practical work (Practice).

Then, the evaluation method would be:

\[ 0.6 \times \text{Theory} + 0.4 \times \text{Practice} \]

where:

\[ \text{Theory: } \max(T2, 0.5 \times T1 + 0.5 \times T2) \]

**Teamwork:**

Evaluated using a simple rubric that each group tutor group uses to rank different aspects of teamwork of every member of the group.
BIBLIOGRAPHY

Basic:
- DeNero, John. Composing Programs.

RESOURCES

Hyperlink:
- [https://cs61a.org/](https://cs61a.org/)
- [https://docs.python.org/3/reference/index.html](https://docs.python.org/3/reference/index.html)