Universitat Autònoma de Barcelona

**2023/2024**

# Artificial Intelligence

Code: 102768
ECTS Credits: 6

| Degree | Type | Year | Semester |
|---|---|---|---|
| 2502441 Computer Engineering | OB | 2 | 2 |

## Contact

Name: Maria Isabel Vanrell Martorell

Email: maria.vanrell@uab.cat

## Teaching groups languages

You can check it through this link. To consult the language you will need to enter the CODE of the subject. Please note that this information is provisional until 30 November 2023.

## Teachers

Pau Folch Codera

Mehmet Anil Akbay

## Prerequisites

Given the introductory nature of this subject, we assume the student does not have any previous knowledge about this topic. Is the aim of the subject to give to the students the means to acquire the knowledge contents described in the contents section of this guide.

About other skills we expect from the students:

- Having coursed a technological or scientific background in the secondary school.
- Having passed the programming subjects of the 1st and 2nd year of this degree.
- Knowing basic notions of python programming
- Having skills at a user level of one of the following platforms (Windows, Mac or Linux)
- Having access to a computer, and much better if it is a laptop

## Objectives and Contextualisation

This is a subject devoted to introducing the Student to the artificial intelligence (AI) field. Two main objectives are pursuit:

a) let the students to learn how resolution of AI problems is performed with their own specificities in representation, evaluation and solving methodologies, and

b) let the students to know a wide range of techniques and basic algorithms that allow to solve the proposed problems and improving their programming skills.

More specifically, these two aims are pursuing:

- Giving a basic historical introduction and aims of the AI field
- Facing the student with the problem of selecting a good knowledge representation as the basic step to solve an AI problem
- Familiarize the student in 4 different knowledge representations and their corresponding algorithms.
- Giving to the student the ability to design solutions to contextualized problems
- Giving to the student the ability to present and justify the adequacy of the designed solutions.

## Competences

- Acquire personal work habits.
- Acquire thinking habits.
- Have the capacity to conceive, develop and maintain computer systems, services and applications employing the methods of software engineering as an instrument to ensure quality.
- Have the capacity to conceive, draft, organise, plan, develop and sign projects in the field of computer engineering for the conception, development and exploitation of computer systems, services and applications.
- Know and apply the basic algorithmic procedures of computer technologies to design solutions for problems and to analyse the adequacy and complexity of the algorithms proposed.
- Know and apply the fundamental principles and basic techniques of intelligent systems and their practical applications.
- Knowledge, design and efficient use of the most suitable data types and structures for problems solving.

## Learning Outcomes

1. Apply different software methods, architectures, concepts and patterns in the design, development and analysis of computer programs.
2. Demonstrate a high capacity for abstraction.
3. Design, develop, select and evaluate applications, ensuring their reliability and security.
4. Identify and define the computational solution for pattern recognition, and decision-making based on the exploration of alternatives, reasoning with uncertainty and task-planning.
5. Identify the computational complexity of an algorithm in terms of memory resources and run time.
6. Select and apply the most appropriate combination of data structures and strategies to resolve a computing problem.
7. Work independently.

## Content

1. Introduction to AI. Aim and brief history of the field. Definition of rational agent, methodologies for solving AI problems and knowledge representation.

2. Problem solving by Searching on alternatives

2.1. Informed Search. Basic definitions on search and review of non-informed search algorithms. Analysis of search algorithms: Completeness, Optimality and Complexity. Heuristic concept and examples. Informed Search: basic and optimal. A* and its properties. Effective branching factor and heuristic properties.

2.2. Local Search. Basic definitions, pros and cons. Understanding local search as exploring the heuristic function landscape. Local search with known goal: Hill-Climbing algorithm. Problems of local search: local maxima, plateaus and ridges. Local search with unknown goal: Steepest Ascent, Steepest Ascent with local maxima control, Simulated Annealing.

2.3. Adversarial Search with Minimax. Basic definitions. Minimax algorithm. Examples. Alpha-beta pruning. Complexity analysis. Minimax improvements: progressive deepening, singular extension heuristic (horizon effect). Examples of heuristic functions.

2.4. Adversarial Search with Random Simulations. Basic definitiions. Monte-Carlo Tree Search algorithm. Examples.

3. Solving pattern recognition problems.

3.1. Statistical approaches. Feature spaces as a representation for case-based reasoning. Basic definitions on feature spaces. Feature selection and dimension reduction. Classification based on supervised learning: decision functions, assumptions about the data distribution, local search to find decision function, k-nearest neighbour functions. Classification based on unsupervised learning: K-means algorithm, searching the best k with Fisher discriminant. Analysis of the algorithms.

3.2 Structural approaches. Graphs and semantic networks. Basic definitions and representation with adjacency matrices. The graph matching problem, basic algorithms, improvements (AC4) and complexity. Inexact graph matching: similarity measures, edition distance. Study case: String-matching.

4. Solving problems of reasoning

4.1. Logic and inference mechanisms. Knowledge representation: propositional logic and predicate logic. Review of basic algorithms: natural deduction, resolution mechanism, unification, clausal form conversion. Basic definitions and algorithms on rule-based Systems: rule base, working memory, rule chaining mechanisms, conflict resolution strategies.

## Methodology

Artificial Intelligence is defined by the type of problems is trying to solve, thus in this course, the type of problem is organizing the course content. We will work in three different types of sessions:

Theoretical session: These sessions are classical lectures based on the lecturer explanations, motivating the students to participate in order to ensure they are achieving the knowledge transmission.

Problem-based session: These sessions are with a more reduced number of students to facilitate interaction. In these sessions we pursuit to reinforce the understanding of the topics presented in theoretical sessions by posing practical cases that require the design of a solution using the methods presented in theory. It is impossible to follow these sessions without following theoretical sessions, since they are strongly linked. In these sessions we do interactive quizzes to evaluate the participation and the the achievements of the students.

Practical session: It is the type of session where different activities are performed connected to perform individual and team-based projects. Different kinds of activities are done in these sessions: (a) sessions for team work but tutored by teaching assistants, (b) sessions to individually evaluate through quizzes to the students on site, (c) sessions to present the results, where all the team members must explain and defend the results of the developed project.

Transversal skills we work in this subject are the following:

- Acquiring thinking skills, specifically developing systemic thinking.

This skill is acquired from the theoretical contents, since all the contents are based on building intelligent system with interacting modules, as well as from the projects the students work on, since they work on teams that generate a lot of dependencies in their tasks. This skill is evaluated in the theoretical exams, in the problem task delivered and at all thelevels of the evaluation of the project.

- Acquiring personal work skills, specifically autonomous working.

This skill is acquired from the study of the theoretical contents, the individual delivery of the problem assignments and by the student participation in the project. In the three cases the individual work is evaluated, in theoretical contents through the exam, in problems through the evaluation of the deliverables and in the project through the individual quizzes, the participation in the final presentation and the intra-evaluation performed by the team members within each group.

Annotation: Within the schedule set by the centre or degree programme, 15 minutes of one class will be reserved for students to evaluate their lecturers and their courses or modules through questionnaires.

## Activities

| Title | Hours | ECTS | Learning Outcomes |
|---|---|---|---|
| Type: Directed | | | |
| Theoretical lectures | 30 | 1.2 | 2, 4, 5 |
| Type: Supervised | | | |
| Practical sessions | 12 | 0.48 | 1, 4, 6 |
| Problem-based sessions | 10 | 0.4 | 2, 4, 5, 7 |
| Type: Autonomous | | | |
| Individual study | 38 | 1.52 | 2, 4, 7 |
| Practical work on project | 51 | 2.04 | 1, 2, 3, 5, 6, 7 |

## Assessment

To evaluate the learning level of the students we establish a formula that combines acquisition of knowledge, abilities in problem solving and the skills to work in a team as well as to justify the obtained results in a project.

Final grade is computed as a weighted sum of the marks obtained in the different performed activities:

***Final Score = 0.5 * Theory Score + 0.1 * Problem Score + 0.4 * Practice Score***

This formula will be computed only in the case that Theory Score is greater or equal than 5, and the Project Score is greater or equal than 6. No constrain is applied on the Problem Score. If the Final Score is >=5, but the minimum thresholds are not achieved in on the the Scores, then the Score in the transcript will be 4.5.

The Theory Score is calculated as the average of the score of two Partial Exams:

***Theory Score = 0.5 * Partial 1 Score + 0.5 * Partial 2 Score***

this mark is computed only if the Partial Scores are both equal or greater than 3.5. When the Scores are given, a date and time of an Exam Review Session will be announced.

Retake Exam. In case the Theory Score does not achieve the 5 to pass, the student can retake the exam 1, the exam 2 or both. To compute the final Theory Score we consider the maximum between the first mark and the corresponding retake mark.

The Problem Score pursuits the student to work on the theoretical content as they are given during the course, specific problems are posed to make the student have to apply the theoretical contents just after they have been explained in the lectures. To evaluate this activity students are asked to do a weekly delivery of solved problems, and the active participation of the students in the weekly Problem Sessions.

***Problem Score = 0.7 * Deliveries Score + 0.15 * Session Attendance + 0.15 * Session-Quiz Scores***

this score is included in the Final Score if the % of deliverables is superior to 70% of the total.

The Practice Score has an essential weight in the final mark, it pursuits the student to program and explore the studied techniques within the frame of the global goal of a contextualized project. Additionally, the student has to demonstrate its skills in doing all this work both, individually and in a team, and defending the final results in a presentation. The final Project Score is computed as:

***Practice Score = 0.5 * Project 1 Score + 0.5 * Project 2 Score***

To compute this Score every one of the projects has to get a mark greater or equal than 6. The mark for each one of the projects is computed from a compilation of marks:

***Project 1 Score =*** *0.6 * Code Mark + 0.5 * Quiz Mark*

- *Code Mark: is derived from the evaluation of the delivered code with some test codes. These tests are prepared to evaluate correctness and efficiency.*
- *Quiz Mark: is evaluated with an exam where the student can access own code on personal laptop.*

Code Mark and Quiz Mark must be greater or equal than 5.

***Project 2 Score =*** *0.75 * Group Mark* + 0.25 * Individual Mark

- *Group Mark = 0.6 * Code Mark + 0.3 * Report + 0.1 * Group Presentation*
- *Individual Mark = 0.5 * Individual Presentation + 0.5 * Group Participation*

The Code Mark and the Individual Mark must be greater orequal than 5.

Retake Sessions. In case a Project Score does not arrive to the adequate level to pass, the students will have a retake option. In the case the student needs to retake the Individual Mark of Project 2, this will be made with a Quiz similar to the Project 1 Quiz. In case the student requires to retake any part of the a Project Mark, then the final Project Score will be 7 as the maximum.

Single Assessment

This evaluation consists of the following activities:

- Activities to evaluate the Theory: student has to do 2 Partial Exams that are retaken separately (50%, 25% each, 2.5 hours).
- Activities to evaluate the Problems: student has to Deliver 12 Assignements of Solved Problems (10%)
- Activities to evaluate the Practicum: student has to code 2 Projects requiring different evaluation activities:
    1. Code Delivery of Project 1 (12%)
    2. Exam of Project 1 (8%)
    3. Code Delivery Project 2 (12%)
    4. Report Delivery of Project 2 (6%)
    5. Oral Presentation of Project 2 (2%, 20min)

To enable the computation of the final score, some threshold has to be fulfilled by some of the scores:

- Partial Exams 1 and 2 must be greater or equal than 3.5.
- Average score of Exams 1 and 2 must be greater or equal than a 5.
- Project 1 Code mustbe greater or equal than 6.
- Project 1 Exam must be greater or equal than 5.
- Code, Report and Oral presentation of Project 2 must be greater or equal than 5.

Both the retake and the final qualification revision will follow the same system as theContinous Evaluation.

Important notes:

In case the subject is not passed due to one of the evaluation activities does not arrive to the minimum required score, the final score in the transcript will be the minimum between 4.5 and the final mark obtained if the threshold is not considered, with the exception that the numeric score in the transcript will be in between 3.0 and the final mark obtained if the threshold is not considered, for the case the student have performed any irregular act in an evaluation activity, such as those explained below.

Grade will be "Non-Graded" ("No Avaluable") in the case the student did not participate in any of the evaluation activities.

Grade will be "Honours" ("Matrícula d'Honor") in the case the rank of the grade is less than the maximum number of honours can be given in a course, and the value of the grade is over a threshold that will be stablished by the teacher.

The evaluation and delivery dates will be published Campus Virtual cv.uab.cat and might be shifted if there is any change in the planning due to any unexpected event. Students will be informed about any change through cv.uab.cat that will be the usual communication mean between students and teachers.

For the case of students retaking the course, no recognition with grades of the previous year will be considered.

Notwithstanding other disciplinary measures deemed appropriate, and in accordance with the academic regulations in force, assessment activities will receive a zero whenever a student commits academic irregularities that may alter such assessment. Assessment activities graded in this way and by this procedure will not be re-assessable. If passing the assessment activity or activities in question is required to pass the subject, the awarding of a zero for disciplinary measures will also entail a direct fail for the subject, with no opportunity to re-assess this in the same academic year. Irregularities contemplated in this procedure include, among others:

- the total or partial copying of a practical exercise, report, or any other evaluation activity,
- allowing others to copy,
- unauthorized and/or non-cited use of AI tools (such as, Copilot, ChatGPT or equivalent) to solve exercises or projects or any assessed activity,
- presenting teamwork that has not been entirely done by the members of the team,
- presenting any materials prepared by a third party as one's own work, even if these materials are translations or adaptations, including work that is not original or exclusively that of the student,
- having communication devices (such as mobile phones, smart watches, etc.) accessible during theoretical-practical assessment tests (individual exams).

An overall grade of 5 or higher is required to pass the subject. A "non-assessable" grade cannot be assigned to students who have participated in any of the individual partial tests or the final exam.

No special treatment will be given to students who have completed the course in previous academic years, except that the seminar grade previously obtained can be assigned to this course gradebook.

The grade in the Transcript of Records (ToR) will be the lowest value between3.0 and the weighted average grade, in the eventof irregularities having been committed for any assessment activity (and therefore re-assessment will not be possible).

## Assessment Activities

| Title | Weighting | Hours | ECTS | Learning Outcomes |
|---|---|---|---|---|
| Delivery of solved exercises | 0.1 | 0 | 0 | 4, 5, 6, 7 |
| Individual Exam | 0.24 | 6 | 0.24 | 4, 5, 6 |
| Practical project defence (Report+Code+Intra_grupal+Presentation+Quiz) | 0.4 | 3 | 0.12 | 1, 2, 3, 4, 5, 6, 7 |

## Bibliography


- S. Russell i P. Norvig, Artificial Intelligences - A modern approach. Prentice Hall, 2003, http://aima.cs.berkeley.edu/
- Tveter, Donald R., (1998), The Pattern Recognition basis of Artificial Intelligence. IEEE Computer Society.
- Stuart Russell. Human Compatible: AI and the Problem of Control Penguin Publishing Group, Octubre 2019
- Melanie Mitchell. Artificial Intelligence: A Guide for Thinking Humans. Farrar, Straus and Giroux, Octubre 2019.


Interesting videos:

- Documental CODEBREAKER http://www.turingfilm.com/about/overview
- Documental Netflix AlphaGo (2017) https://es.wikipedia.org/wiki/AlphaGo_(pel%C3%ADcula)

## Software

Tools for Programming in Python Language with special attention to the Numpy library