Reinforcement Learning
Mid Semester Project - 2025

The main goal of this mid-semester assignment is to summarize the key topics (theoretical & practical) that we have covered so far in the course integrating some theoretical and practical aspects.

MiniGrid env.

In this exercise you will solve two variations of the "Minigrid" environment.

MiniGrid is a lightweight, 2D grid-world environment with goal-oriented tasks. The agent in these environments is a triangle-like agent with a discrete action space. The tasks involve solving different maze maps and interacting with different objects such as doors, keys, or boxes.

Minigrid documentation:

https://minigrid.farama.org/environments/minigrid/

In this assignment there are two minigrid environments:

1. Random**Empty**Env_10
    a. Solve the environment as is
2. Random**Key**Env_10
    a. Here, you can use up to two reward shaping if you find it helpful. (i.e. you can change the original reward of the env for two events) (For example: I try a reward that seems like: "if door was closed and now it is open, reward = 100" is this is a good reward?)

You can find a demonstration of how to use/render the game (random actions) in the template notebook:

https://colab.research.google.com/drive/1uG0T4chyOF2eISbONVB4Z5xdvXgyPlMn?usp=sharing

a. Show your knowledge in tabular RL that covers what we have learnt so far.
b. Try to “solve” it as fast as you can (less steps\episodes) - this is a competition part.

- Analyze the env. MDP? Episodic? Discrete/continuous action? Discrete/continuous states? Fully/partial observable?
- You need to solve it with different algorithms: MC, SARSA, Q-LEARNING.
- Discuss the advantages/disadvantages of the approaches relating to your minigrid mission.
- Show relevant graphs comparing the different approaches.
- How to represent the state? State size? Q-table size?
- Different hyperparameters: Learning rate, Epsilon…
- Choose discount factor.
- Different initializations.
- Different considerations for the two environments.
- Approaches for Exploration-Exploitation tradeoff.
- Graphs for the training stage and for the inference stage - Average steps to finish episode.
- Evaluate both the good and bad points of your approach.
- Use a specific cell to indicate the best training/inference parameters.

Guidelines:

1. You are not allowed to use an existing RL library. Write the algorithm yourself.

2. You are not allowed to use deep reinforcement learning approaches or RL algorithms that you haven't learnt in the course.

3. For the most successful experiment, in each exercise, Show (In the notebook and report) the number of steps it took the agent to solve the environment, and a convergence graph (Rewards according to the number of episodes). All training outputs will be displayed in the notebook.

4. Add video clips (In the notebook) that shows the agent in the middle of the training process and after the learning completed and converged.

5. Even if you did not fully solve an environment - supply graphs describing the average rewards on X episodes (after finishing the training phase).
   Submit a final report in pdf format when your ID numbers are included in the file name.

6. The final report must be a professional, self-contained scientific document that fully explains your methodology and results without the need to reference the code.

7. You need to supply your code as Google Colab notebooks link. Such that the course team can run it.

8. Write a clean code! Separate code cells and make extensive use of text cells. A sloppy notebook will result in a lower grade.

9. Do not change anything in notebook after the final submission date. A notebook that has been ran/changed after the submission date will be automatically disqualified.

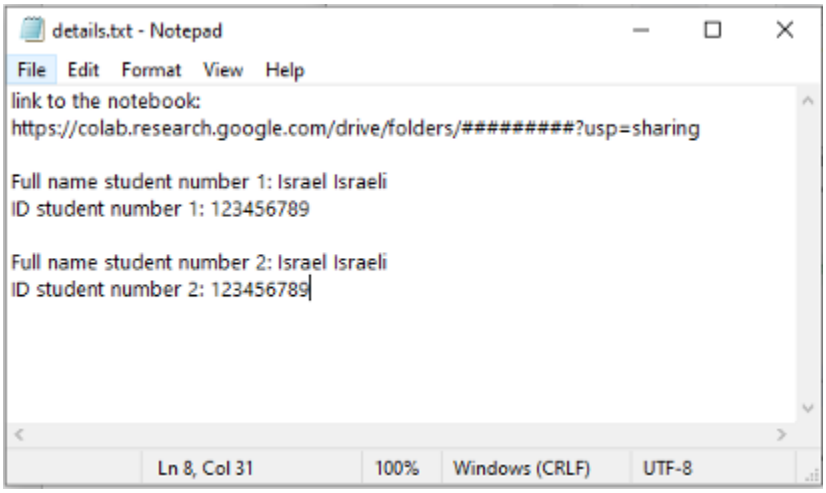
The report will be submitted to the submission box. The file name of the report will contain your IDs as follows: **report_ID1_ID2.pdf**

Do not forget to include the project title, your name and ID in this file.
Max number of pages: 10 (but you don't have to use them all).

The notebook will be presented in an orderly and clean manner; it will contain separate code cells and text cells that explain the actions performed.
** **Very important** ** - when submitting, the notebooks will contain **all the outputs** relevant to the training results.

To the submission box you will also submit an **explainer** file that contains instructions and explanations on how to operate your notebook and other relevant details that need to be known to those who want to use your notebook.

Enter your details in the text file named **details.txt**, the address of your notebook, the names and ID of the two partners, as follows:



To sum up, in the submission box, submit the following files:

1. details.txt
2. report_ID1_ID2.pdf
3. explainer.txt

Team size
The project will be performed in groups of **2 students**.

Academic Integrity

Team/Student may not copy code from other teams/students. Copying answers or code from other students for a project is a violation of the university's honor code and will be treated as such. All suspicious activity will be reported to the head of the department and the university authorities.

Giving code to another student is also considered a violation. Students are responsible for protecting their own work from copying.

If you build some of your code on existing work and utilize existing code (your own or code found on the web), you must give proper attribution to all existing work that you used and make it clear what you changed and contributed. Any unattributed or uncited work that you use will be considered a breach of academic honesty and dealt with according to the course policy in the syllabus.