Question 1: Implement Algorithms for Inverse Reinforcement Learning using Linear Function Approximation [Maximum 25 marks]

Refer to the paper, *Algorithms for Inverse Reinforcement Learning* by *Andrew Ng and Stuart Russel* (Link). You are expected to read the full paper and understand the algorithms that the authors are proposing. In particular, the authors propose three algorithms to implement Inverse Reinforcement Learning (IRL),

- IRL in Finite State Spaces
- Linear Function Approximation in Larger State Spaces
- IRL from Sampled Trajectories

The authors have provided three examples, one for each algorithm, in Section 6. Your task is to reproduce the results shown in the paper for each of the three examples. Below is the overview of each experiment, along with the results presented in the paper.

Note: All the references used in this assignment (i.e., Section x or Figure y) are from the above-said paper. Please refer to the paper to better understand the questions.

1. **IRL in finite state space for *5x5 grid world* [10 marks]**
   In this experiment, a 5x5 noisy grid world is used, for which an optimal policy is given in the paper. The authors applied the algorithm proposed in Section 3.2 for $\lambda = 0$ and $\lambda = 1.05$ and determined the rewards (shown in Figure 2), which are compared with the true reward function shown in Figure 1. You need to perform the following tasks,
   - Design a 5x5 noisy grid world as described in Section 6.
   - Implement the algorithm proposed in Section 3.2. Use the optimal policy given in the example.
   - Solve the linear program mentioned in the algorithm for $\lambda = 0$ and $\lambda = 1.05$, and plot the obtained reward functions for both λ values. Compare these reward functions with the true reward function.
2. **IRL using linear function approximation in larger state space for *mountain-car* [15 marks]**
   In this experiment, mountain-car environment is used. The authors applied the algorithm proposed in Section 4 and determined the rewards (shown in Figure 4(top)). You need to perform the following tasks.
   - Use OpenAI MountainCarContinuous-v0 environment.
   - Implement the algorithm proposed in Section 4 by following the description given in the example.
   - Reproduce the results shown in Figure 4 (top). You are **NOT** required to reproduce the result for the second problem (parking at the bottom of the hill) in this example.

Bonus Question: IRL from sampled trajectories for *continuous version of 5x5 grid world* [25 marks]

In this experiment, a noisy continuous grid world is used, where state space is $[0, 1]$ x $[0, 1]$. The authors applied the algorithm proposed in Section 5 for $\lambda = 0.9$ and determined the rewards. To test the obtained rewards, they compared the obtained reward's optimal policy with the true optimal policy (shown in Figure 5). You need to perform the following tasks.

- Design a continuous noisy grid world as described in Section 6.
- Implement the algorithm proposed in Section 5 by keeping all the required parameters the same as shown in the example.
- Reproduce the results shown in Figure 5.

Submission Instructions

- Submit your source code in main.ipynb, also exported into a main.html. No trained models.
- Clearly indicate at the start which components you have attempted.
- Write suitable comments to describe the methods you have implemented.
- Provide proper attribution to any reference implementations you used.

Plagiarism Policy: Plagiarism detection software is guaranteed to be run before any evaluation. Trying to beat any such software will make your code significantly unreadable and easy to prove malicious intent. In case of heavy plagiarism - all parties involved (giver, taker) will get a 0.