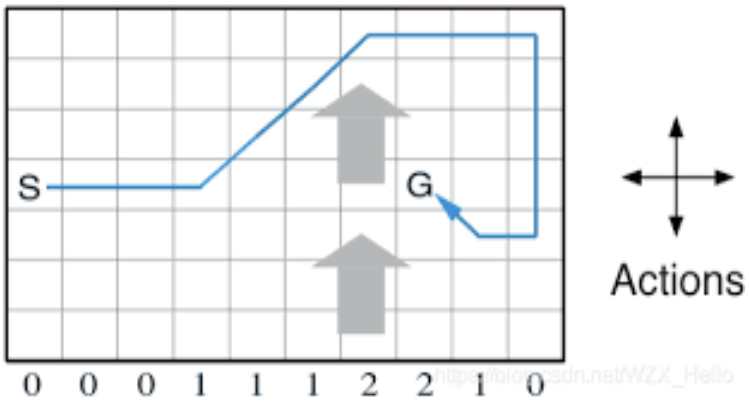
Windy Grid World

ASSIGNMENT 2

NAME: LOMAI MOHAMED SAADELDIN

ID: 20398049



DESCRIPTION

Windy Grid world problem for reinforcement learning. Actions include going left, right, up, and down. In each column, the wind pushes you up a specific number of steps (for the next action). If an action would take you off the grid, you remain in the previous state. For each step, you get a reward of -1, until you reach a terminal state.

METHODOLOGY

Part 0: Import Libraries

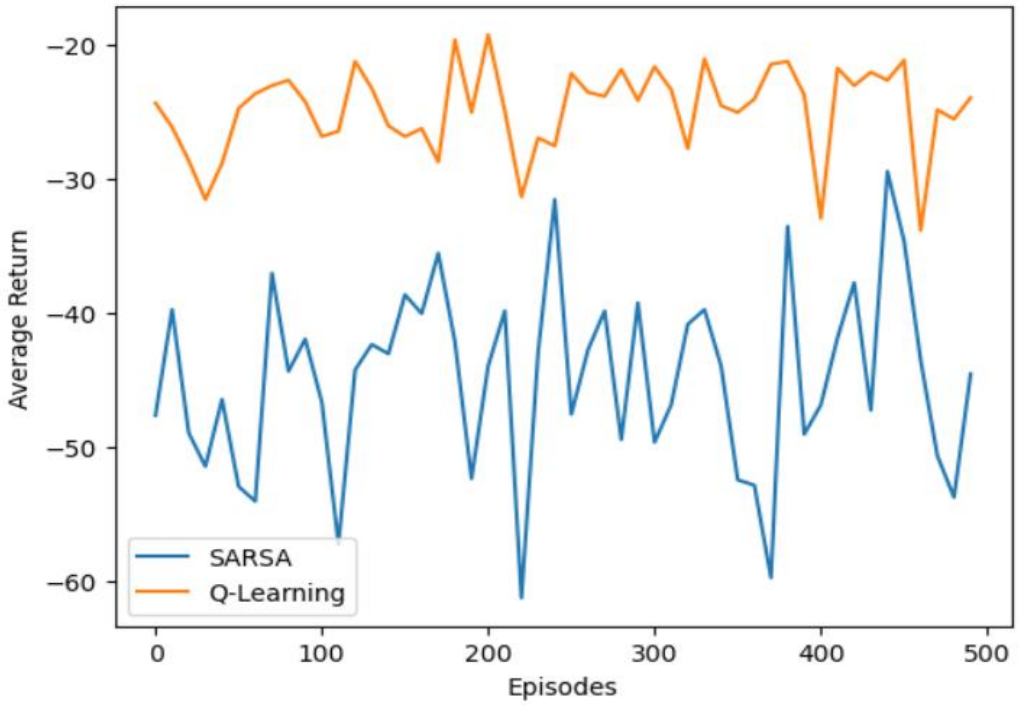
We begin by importing the necessary packages.

Part 1: Build Windy Grid World environment.

The WindyGridworld environment class is a class in reinforcement learning that represents a gridworld environment with wind. It has four methods: init, reset, step, and render. The init method initializes the environment parameters such as the size of the grid, starting and goal positions, wind strength, diagonal movement, stochastic wind, and penalize movement. The reset method resets the environment to its initial state and returns it. The step method takes an action in the environment and returns the next state, reward, and done flag. The render method renders the current state of the environment. The class can be initialized with several parameters such as the grid shape, starting and goal positions, wind strength, and movement options. The default values for these parameters are provided.

- The output "Average return for SARSA policy: -34.0" indicates that, on average, the SARSA policy obtained by running the SARSA algorithm with the specified hyperparameters achieves a return of -34.0 for a single episode in the Windy Gridworld environment.
- This output suggests that the SARSA policy is not performing well on the Windy Gridworld problem, since the average return is negative and relatively far from the optimal return of 0.0. It may be necessary to adjust the hyperparameters or try a different algorithm to improve the performance of the agent on this problem.

- The output "Average return for Q-learning policy: -22.0" indicates that, on average, the Q-learning policy obtained by running the Q-learning algorithm with the specified hyperparameters achieves a return of -22.0 for a single episode in the Windy Gridworld environment.
- Compared to the SARSA policy with an average return of -34.0, the Q-learning policy appears to be performing better, since the average return is less negative and closer to the optimal return of 0.0. This suggests that Q-learning may be a more effective algorithm for the Windy Gridworld problem than SARSA, at least with the specified hyperparameters.



- The resulting figure shows the learning curves for both algorithms on the same plot. The figure shows that both algorithms start with low average returns and gradually improve over time. However, Q-learning appears to converge to a higher average return than SARSA, suggesting that it may be a more effective algorithm for the Windy Gridworld problem.

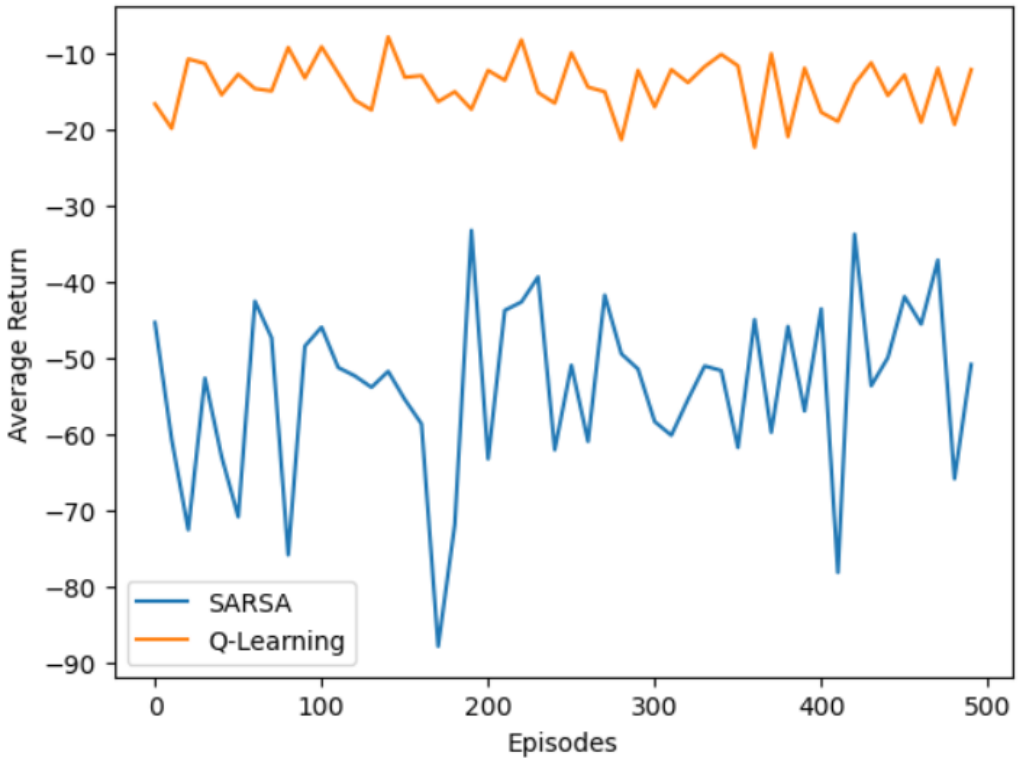
Part 3: Build Windy Grid World environment with King's Moves and Stochastic wind:

This code represents a different implementation of the Windy Gridworld environment in reinforcement learning. The main differences between this code and the previous environment are:

1. The action space is defined as a discrete set of 4 actions (up, right, down, left) instead of 8 actions (up, up-right, right, down-right, down, down-left, left, up-left).
2. The state space is the same, defined as a list of all possible states in the gridworld.
3. The wind pattern is defined as a list of 10 integers, one for each column of the gridworld, instead of a list of 8 integers, one for each diagonal of the gridworld.
4. The next state is computed based on the selected action and the wind effect using if-else statements instead of a lookup table.
5. The reward and done flag calculation is the same, based on the next state.
6. The render function is the same, used to render the current state of the gridworld.

- The output "Average return for SARSA policy: -18.0" indicates that the average return obtained by the SARSA policy on the WindyGridworldKing environment was -18.0 over one episode. This means that, on average, the policy achieved a total reward of -18.0 when navigating the gridworld from the start to the goal state. The negative value indicates that the policy did not perform well, as it received more penalties than rewards during the episode.

- The output "Average return for Q-learning policy: -15.0" indicates that the average return obtained by the Q-learning policy on the WindyGridworldKing environment was -15.0 over one episode. This means that, on average, the policy achieved a total reward of -15.0 when navigating the gridworld from the start to the goal state. Compared to the SARSA policy, the Q-learning policy obtained a higher average return, which suggests that it performed better in this environment.



- The resulting figure shows the learning curves for both algorithms on the same plot. The figure shows that both algorithms start with low average returns and gradually improve over time. However, Q-learning appears to converge to a higher average return than SARSA, suggesting that it may be a more effective algorithm for the Windy Gridworld problem with King's Moves and Stochastic wind.