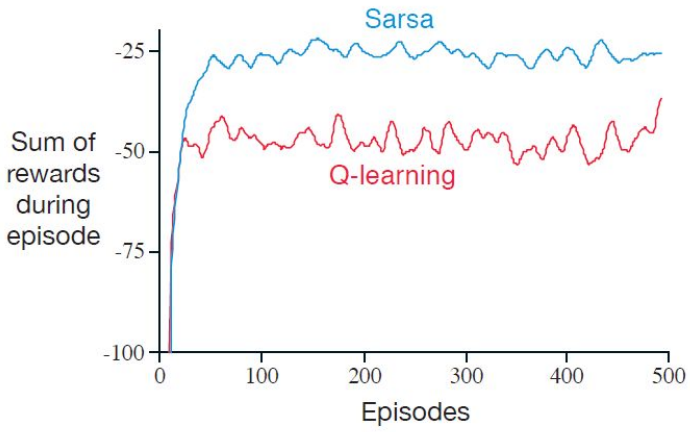
Assignment 3

Implementation of Monte Carlo and Temporal Difference Learning Algorithms
Due: Monday, March 1st by 11:59pm

For each item, provide a writeup of your findings and your experience. What did you struggle with? What did you learn?

1a. Recreate the figure below in the text using the 'CliffWalking-v0' task in OpenGym.



1b. Add on-policy Monte Carlo. How does it compare?

2a. Implement <u>one</u> **of the following to learn the Frozen Lake task in OpenGym.** Don't worry if you can't solve it right away. Frozen Lake is a "toy" task, but solving it without changing the reward function is non-trivial.

1. Double Q-Learning
2. Expected SARSA

2b. What is the reward function for the Frozen Lake task? How does this affect your agent's ability to learn? How might you make things easier for your agent? Modify the reward your agent receives from its actions. Can you achieve better performance?

- *env.desc gives you the map. 'H' is a hole, 'G' is the goal, 'S' is the start, and 'F' is the frozen lake.*

2c. Now let's achieve better performance without cheating. Frozen Lake is about exploration. How could you promote better exploration? What's the best performance you think you can get, and why?

- *You might try a different initialization approach, decay epsilon over time, make your agent wait until to update|decay until it has received a reward of some kind, etc.*

3. Implement tabular Q-Learning to solve the Mountain Car task in OpenGym.

- Note that the state is a length 2 vector and is *continuous.* You will have to discretize the state to solve the task. How you discretize the state is up to you.