2/1/17
580-03, Spring 2017, Programming Project 1

In this homework, you will implement a dynamic programming method. I recommend you use Open AI Gym, as discussed in class because Gabe is very familiar with this platform. However, if you would like to use a different framework (e.g., BURLAP, RL-Glue, RL-PI, etc.) you can. We just won't be able to help as much.

First, install the OpenAI Gym: https://gym.openai.com/
Then download the code for this project from github:
https://github.com/IRLL/reinforcement_learning_class
mdp_gridworld.py model a version of the 12-state gridworld in the Isbell & Littman videos.

Recall that homework is graded on a 10-point scale. Here is a guide to how we will grade your homework. To get a 7/10, you only need to successfully implement policy iteration or value iteration and show that it works.

Ideas for making your report more interesting, with rough point values.

+1 point: Empirically compare how long learning takes, and the policy produced, when you change the rewards and/or the discount factor in the task.
+0.5 point: Implement both policy iteration and value iteration. Empirically evaluate the difference.
+0.5 point: Show how the policy and the policy's return changes over each iteration. Compare with a policy that acts completely randomly.
+0.5 point: Show how the policy can be learned faster/slower and that the algorithm converges to a better/worse policy when changing the Δ parameter.
+1.5 point: Use a different task, or modify the task provided, to show that your method works with stochastic actions (i.e., a non-deterministic transition function). Use two different settings: one where things are only slightly random (e.g., 95% chance of moving in direction selected and 5% chance of moving parallel to direction selected) and one where things are very random (e.g., 50% chance of moving in selected direction, 40% chance of moving in one of the parallel directions, 10% chance of moving backwards). What is the new policy that's learned? Does it take more/less time to learn it?
+1.5 point: Implement an asynchronous dynamic programming method. Modify your algorithm so that it doesn't always have to do a full sweep of the state space in the same manner each time. Does it help or hurt the performance? Why?