EECS 649: PROBLEM SET #12

Reading:

- R&N Chapter 21.1-4,6; 17.5
  You can do PS#12 with this reading and the Reinforcement Learning (RL) and Game Theory (GT) lectures.

  Specifically:
  - Problems 12.1-4 and the program 12.7 can be done after the RL lecture.
  - Problems 12.5 and 12.6 are on material in R&N 18.2 and the GT lecture.

Total Points: 100

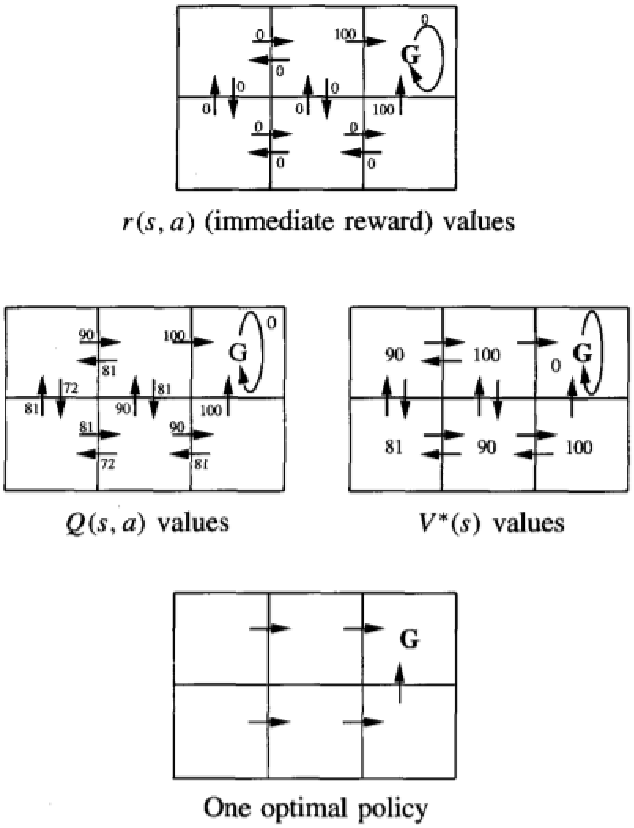
Notes:

- Submitted electronically (via Gradescope)
- ***** This problem set has an **extensive FAQ** linked here *****

Problem 12.1 [10 points]
Please type your answer to this question.
Imagine that you are designing a robot to run a maze. You decide to give it a reward of +1 for escaping from the maze and a reward of zero at all other times. The task seems to break down naturally into episodes--the successive runs through the maze--so you decide to treat it as an episodic task, where the goal is to maximize expected total reward $r_1 + r_2 + ... r_N$, where N is the length of the episode. After running the learning agent for a while, you find that it is showing no improvement in escaping from the maze. What is going wrong? Have you effectively communicated to the agent what you want it to achieve?

[Problem 12.2 begins on the next page]

Problem 12.2 [10 points]
Consider the deterministic MDP illustrated in the figure below. Find a second optimal policy. **Draw** your policy on top of a grid like that pictured. How many optimal policies are there?



$r(s, a)$ (immediate reward) values

$Q(s, a)$ values

$V^*(s)$ values

One optimal policy

FIGURE 13.2
A simple deterministic world to illustrate the basic concepts of Q-learning. Each grid square represents a distinct state, each arrow a distinct action. The immediate reward function, $r(s, a)$ gives reward 100 for actions entering the goal state G, and zero otherwise. Values of $V^*(s)$ and $Q(s, a)$ follow from $r(s, a)$, and the discount factor $\gamma = 0.9$. An optimal policy, corresponding to actions with maximal Q values, is also shown.

Problem 12.3 [10 points] ← **This problem has an FAQ**
Both parts refer to the golf example from the Reinforcement Learning lecture.

a. Draw the contours of the optimal state-value function, V*(s).
b. Draw the contours of the optimal action-value function for putting, Q^*(s,putt).

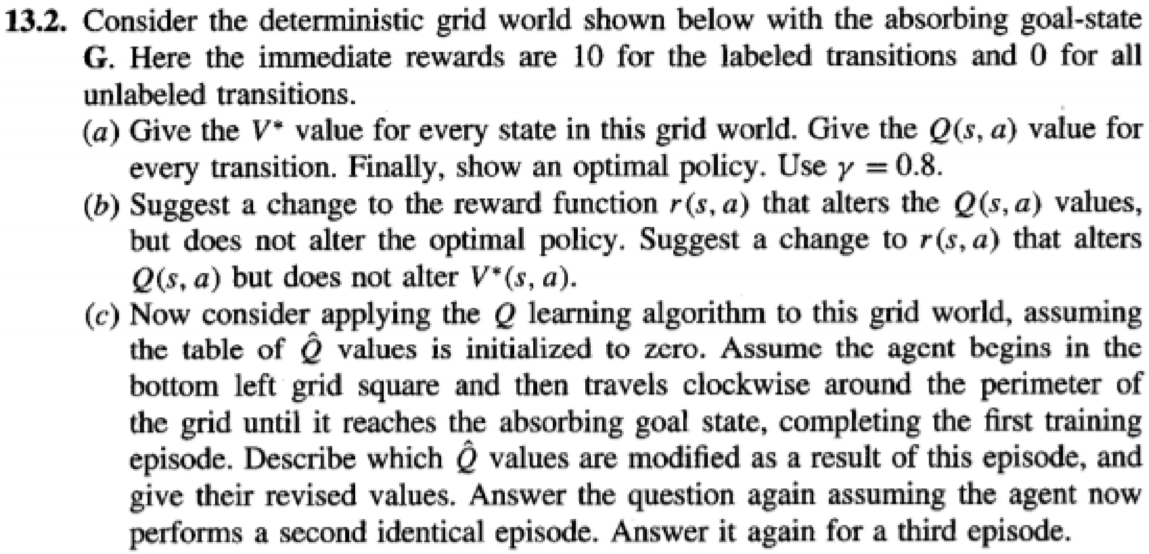
Recall Q*(s,a) is value of choosing action a in state s and then acting optimally thereafter

The cost is −1 for each action/stroke and you may assume that gamma=1 (both just as in golf).

Use this inked sheet for recording your drawings. **Hint**: You can probably see it yourself, but if you get stuck on part (a), we discussed it in a previous lecture.

Problem 12.4 [20 points] ← **This problem has an FAQ**

Do the following, parts a and c **only** (10 points each). For part (a), record your answers on a grid similar to how V* is shown in Figure 13.2 in Problem 12.2 above.

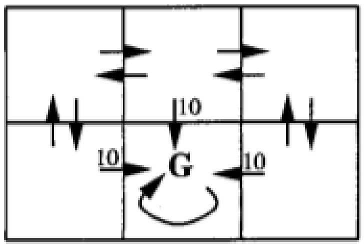


13.2. Consider the deterministic grid world shown below with the absorbing goal-state **G**. Here the immediate rewards are 10 for the labeled transitions and 0 for all unlabeled transitions.

(*a*) Give the V^* value for every state in this grid world. Give the $Q(s,a)$ value for every transition. Finally, show an optimal policy. Use $\gamma = 0.8$.

(*b*) Suggest a change to the reward function $r(s,a)$ that alters the $Q(s,a)$ values, but does not alter the optimal policy. Suggest a change to $r(s,a)$ that alters $Q(s,a)$ but does not alter $V^*(s,a)$.

(*c*) Now consider applying the Q learning algorithm to this grid world, assuming the table of $\hat{Q}$ values is initialized to zero. Assume the agent begins in the bottom left grid square and then travels clockwise around the perimeter of the grid until it reaches the absorbing goal state, completing the first training episode. Describe which $\hat{Q}$ values are modified as a result of this episode, and give their revised values. Answer the question again assuming the agent now performs a second identical episode. Answer it again for a third episode.



Problem 12.5 [10 points] ← **This problem has an FAQ**

Consider the traditional Rock-Paper-Scissors Game.

a. Construct the matrix of pure strategies (+1 = win, -1 = loss, 0 = draw).
b. Show that the mixed strategy of choosing each outcome equally at random produces a saddle solution.

Problem 12.6 [10 points] ← **This problem has an FAQ**

Compute the minimax strategy for an arbitrary "Penny Matching Game," with payoffs as follows:

	H	T
H	a	b
T	c	d

a. If *a*=$1 million, *b*=*c*=-1 cent, *d*=1 cent, with what probability should each player choose "H"?
b. Under what conditions on *a, b, c, d* is the minimax strategy "pure"? (You may have to reason about "edge effects"; you will then have analyzed all 2-person, 2-strategy, zero-sum games.)

The following problem requires some programming.

Problem 12.7 [30 points] ← **This problem has an FAQ**
In this problem, you will be using (passive) **direct utility estimation** to learn a policy evaluation for the 4 x 3 world shown in Figure 17.1 of R&N, 4e. (FYI, others, like Sutton & Barto call this "every-visit Monte Carlo (MC).")

a. First, run your program on the optimal policy in Figure 22.1(a) of R&N, 4e, which uses gamma=1 and R=−0.04. Check your answers against Figure 22.1(b) of R&N, 4e. Record your answers as noted below. Comment on the number of trials used. There is **no need** to turn in code for this part.
b. Next, modify your program to learn the value of the "random policy" (again with gamma=1 and R=−0.04), which in each non-terminal state chooses among the actions *Up, Down, Left, Right* with equal probability. Turn in your code and record your answer as noted below and comment on the number of trials used. Turn in **only** those **portions** of your code that **differ** from any of that provided in linked Python notebook mdp.ipynb or the book's code directory, e.g., its Python MDP code. Specifically, do not turn in the function computing T(s,a,s'), which in this case can be used to simulate the system.

Note: For ease of grading, enter your learned utility values from parts (a) and (b) in the top and bottom grids on this linked sheet, respectively. Caption the drawings appropriately and comment on your results.

Wow, you did it! Twelve challenging problem sets. Congratulations!