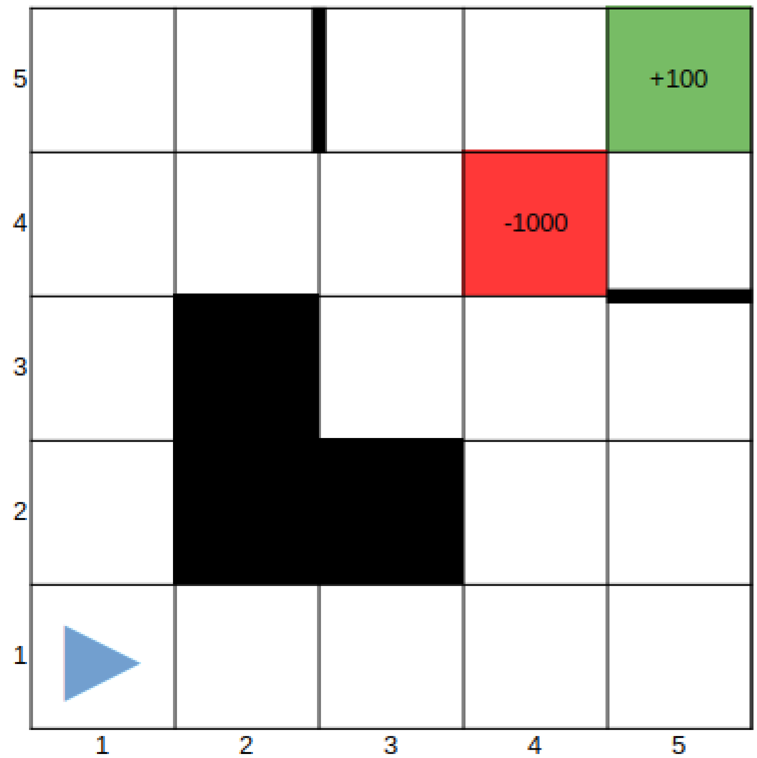
CPSC 4420/6420: ARTIFICIAL INTELLIGENCE

ASSIGNMENT 2

NAME:

Consider the following puzzle. The green and red states are both terminal states, with the rewards as shown (so we can consider the green state the "goal", and the red a "game over" state with a large negative reward). Thick borders between cells represent walls that the robot player cannot cross, and the black squares contain obstacles and cannot be entered. The robot player is represented by the blue triangle, and the direction the triangle points is the way the robot is facing.



Let's represent a state with (x,y,d), where x and y represent the horizontal and vertical positions (i.e. location), and d represents the direction the robot is facing (1: up, 2: down, 3: left, and 4: right).

The robot can take the following actions:

- A_1: Move one cell forward in the direction it is facing. Cost: 1.5
- A_2: Move two cells forward in the direction it is facing. Cost: 2
- A_3: Turn to its left, and stay in the same cell. Cost: 0.5
- A_4: Turn to its right, and stay in the same cell. Cost: 0.5

Note that each action has a different cost value. This can also be considered an immediate negative reward. For example, we have $R(s,A_1,s') = -1.5$. The cost is evaluated on the current state, (the state the robot is in when it begins the action, not the one it lands on after performing the action). In the same way, the value of state V(s) represents the value of the current state and you should initialize the algorithm with $V_1(5,5,x)=+100$, $V_1(4,4,x)=-1000$ (for x=1,2,3,4 representing the robot orientation/direction), and zero for all other states.

So, for example, if the robot is in state (4,1,4), it means that it is in location (4,1) and facing right. The result of possible actions for this state are as follows:

A_1 (move 1 cell forward) --> (5,1,4)
A_2 (move 2 cells forward) --> impossible remains in the current state (4,1,4)
A_3 (turn left) --> (4,1,1) : the robot stays in (4,1) but now faces up
A_4 (turn right) --> (4,1,2) : the robot stays in (4,1) but now faces down

A move is impossible if it would result in landing on a blocked cell, like (2,2), (2,3), or (3,2), or if it would result in crossing a barrier, like moving from state (2,5) to (3,5), or (5,3) to (5,4). A move that would take the robot outside of our 5x5 grid is also impossible.

Note that we have more states than the number of cells, because the robot facing a different direction produces a new state, even if it does not change location. In the example above, if we move to (4,1,1), where the robot is facing up, this is a different state from the one we were in, (4,1,4), even though the robot has not moved cells.

A) If there is no living reward/penalty, no noise, and no discount (gamma = 1), use your common sense to find the best possible route from (1,1) to (5,5).

B) With no discount (gamma = 1), no living reward, and no noise, use the Value Iteration Algorithm with 100 iterations to update the optimal values for each state and print the result [only for the first 10 iterations] in the following format:

iter 1:
state (1,1,1) V = (some value) Best Action: A_i (where i is some number 1-4)
state (1,1,2) V = (some value) Best Action: A_j
…
state (5,5,4) V = (some value)

iter 2:
state (1,1,1) V = (some value) Best Action: A_i (where i is some number 1-4)
state (1,1,2) V = (some value) Best Action: A_j
…
state (5,5,4) V = (some value)

If two actions are tied for best, you can select one at random or always choose the one with the smallest index.

C) If you start from state (1,1,4) and follow the optimal policy you found in part B, does it follow the same path you proposed in part A?

D) Repeat part B with the same assumptions, except for gamma = 0.8 (discount factor). Compare the results with that from part B. Do they match?

E) Repeat part B with the same assumptions, except for gamma = 0.2. Compare the results with that from parts B and D. Do they match?

F) **(Optional for 4420)** Repeat part B, but this time with noise = 0.2, and gamma = 0.9 and no living reward. With a noise of 0.2, every time you take an action, the result will be the expected action with Probability 0.8 (80%), but 20% of the time, the robot will instead take a different action (taken randomly out of unexpected actions, with equal probability). If the action is impossible, it remains in the same cell.

For example, if we are in state (4,1,4), location (4,1) and facing right, and we take action A_1 (moving one cell forward), the resulting state will be:

s'= (5,1,2) with probability 0.8 [because A_1 is rendered]
s'= (4,1,4) with probability 0.2/3 [renders A_2 which is impossible]
s'= (4,1,1) with probability 0.2/3 [because A_3 is rendered]
s'= (4,1,2) with probability 0.2/3 [because A_4 is rendered]

Compare the results with that of the previous parts and explain your observations.