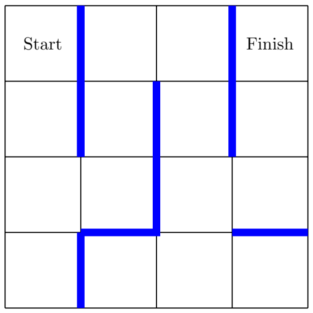
Markov Decision Processes Workshop **CS 480**

Write your answers below. When you are finished, turn this page in, but also share your Python code with me. If you are using Google Colab, share your code with: `lins.brian@gmail.com`. *Each of these problems is easier if you use the MDP class on my website (see the notes for this week).*

1. **Maze runner.** Consider the following maze. The agent begins in the Start square and each round can move one square left, right, up, down as long as there is no barrier blocking its movement and it doesn't move off the grid.



Suppose you use a MDP with a reward of -1 for every state except the Finish state. What are the optimal values (total rewards) that the agent can earn from each state (without discounting)? Put your solution in the grid above. Hint: You don't need a computer for this one.

2. **Dice game.** You are allowed to roll a fair (6-sided) die as many times as you want. After each roll you can decide whether you want to roll again or quit the game. If the die ever lands on a one, the game is over and you receive no reward. If you decide to quit the game before that happens, then you get a reward equal to the sum of all prior rolls, up to a maximum of 100.

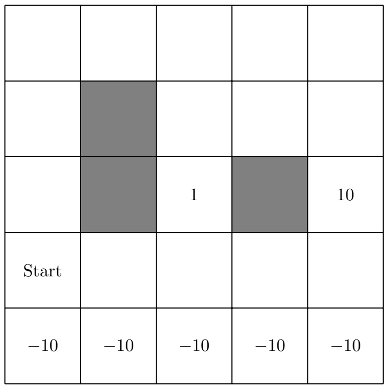
   (a) Implement a MDP in Python to find the optimal policy for this game. Optional: the states can be tuples of the form `(total, finished)` where `total` is the total you have rolled so far and `finished` is 0 if you have not cashed out, and 1 if you have. Note: You won't receive any reward until you cash out, and then you can only receive the reward once, so you might want to use the state `(0,1)` as the final game over state.

   (b) Describe the optimal policy with a simple one sentence rule about when you should cash out.

   (c) What is the expected value of your reward when you first start this game?

3. **Don't fall off the cliff.** The grid below shows the states the agent can enter. The numbers are rewards (states without numbers have a reward of zero). The states with numbers are absorbing and the agent is stuck there forever if it reaches one. The gray squares cannot be entered.

   In the other states, the agent can choose any direction (up, down, left, or right). There is an 80% chance it will move in the direction it picks. There is also a 10% chance it will move in either of the two directions that are perpendicular to its choice. If the agent attempts to step off the grid or into one of the gray squares, then it stays in its current square.



(a) Implement this process as an MDP in Python. You'll need to change the MDP class on my website by adding the option of a discount factor.

(b) What is the expected value of this MDP in the start state with a discount factor of $\gamma = 0.9$?

(c) Which action should the agent choose in the start state?