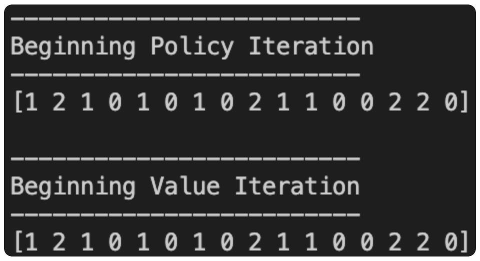
DRL Homework1

2. Dynamic Programming

(a) and (b) can be seen in the file dp.py.

(c)

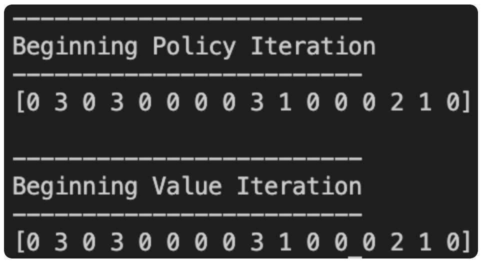
For FrozenLake-v1, we get:



```
-------------------------
Beginning Policy Iteration
-------------------------
[1 2 1 0 1 0 1 0 2 1 1 0 0 2 2 0]

-------------------------
Beginning Value Iteration
-------------------------
[1 2 1 0 1 0 1 0 2 1 1 0 0 2 2 0]
```

For SlipperyFrozenLake-v1, we run 5 times, get the results are same:



```
-------------------------
Beginning Policy Iteration
-------------------------
[0 3 0 3 0 0 0 0 3 1 0 0 0 2 1 0]

-------------------------
Beginning Value Iteration
-------------------------
[0 3 0 3 0 0 0 0 3 1 0 0 0 2 1 0]
```

It seems that the stochasticity do not affect the resulting policy.

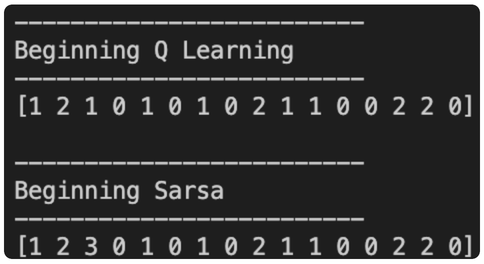
If we take the SlipperyFrozenLake-v1's transition function to be like:

"you have a $\frac{1}{6}$ probability of heading the corresponding direction of your action, and $\frac{2}{3}$ of heading each sideapart from the opposite direction", the thing will change.

3. Temporal Difference Learning

(a)(b):

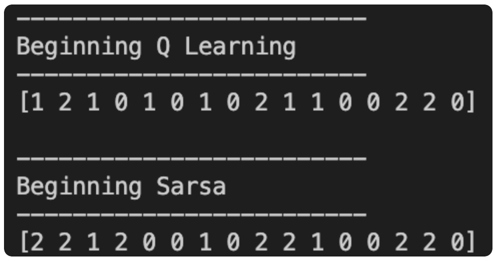
For $\alpha = 0.8, \gamma = 0.9$


```
---------------------------
Beginning Q Learning
---------------------------
[1 2 1 0 1 0 1 0 2 1 1 0 0 2 2 0]

---------------------------
Beginning Sarsa
---------------------------
[1 2 3 0 1 0 1 0 2 1 1 0 0 2 2 0]
```

Do not arrive the terminal.

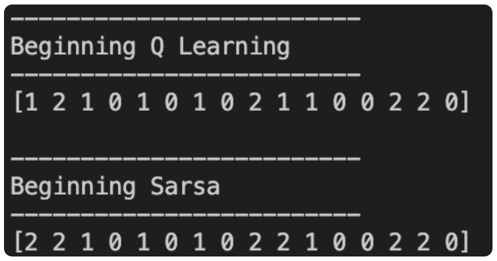
For $\alpha = 0.7, \gamma = 0.8$:


```
---------------------------
Beginning Q Learning
---------------------------
[1 2 1 0 1 0 1 0 2 1 1 0 0 2 2 0]

---------------------------
Beginning Sarsa
---------------------------
[2 2 1 2 0 0 1 0 2 2 1 0 0 2 2 0]
```

Do not arrive the terminal.

For $\alpha = 0.8, \gamma = 0.7$:


```
---------------------------
Beginning Q Learning
---------------------------
[1 2 1 0 1 0 1 0 2 1 1 0 0 2 2 0]

---------------------------
Beginning Sarsa
---------------------------
[2 2 1 0 1 0 1 0 2 2 1 0 0 2 2 0]
```

Do not arrive the terminal.

(c)

An example of a real-world MDP is an autonomous driving vehicle.

Consider a scenario where a vehicle enters an intersection in a city, where each intersection is a state, and the vehicle can choose different directions or actions, such as going straight, turning left or right. These actions make up the action space of the MDP.

The transition dynamics refer to how the state transitions from one state to another when an action is executed in the MDP, the vehicle's state at the next intersection depends on the previous state and the action taken. If the vehicle is currently at an intersection and takes a left turn, it may reach another intersection and continue driving on a new road. To frame this problem as a reinforcement learning problem, we define a reward function and a value function.

We can define a reward function to measure the safety and efficiency of the vehicle before it reaches its destination. We can define a value function to measure the expected return of taking different actions in different states. Reinforcement learning(RL) algorithms can use these functions to learn which actions to take in a given state to maximize expected returns.