Name:Coffman,Ethan
Email: ejc012@uark.edu
ID: 100468386
Late Days: 1

Problem 1:

Iteration 1:

3,3:

1. Up: .6[0+.9*0] + .2[0+.9*0] + .2[1+.9*0] = .2
2. Right: .6[1+.9*0] + .2[0+.9*0] + .2[0+.9*0] = .6
3. Down: .6[0+.9*0] + .2[0+.9*0] + .2[1+.9*0] = .2
4. Left: .6[0+.9*0] + .2[0+.9*0] + .2[0+.9*0] = 0

   New Value : .6

3,2:

1. Up: .6[0+.9*0] + .2[0+.9*0] + .2[-1+.9*0] = -.2
2. Right: .6[-1+.9*0] + .2[0+.9*0] + .2[0+.9*0] = -.6
3. Down: .6[0+.9*0] + .2[0+.9*0] + .2[-1+.9*0] = -.2
4. Left: .6[0+.9*0] + .2[0+.9*0] + .2[0+.9*0] = 0

   New Value : 0

4,1:

1. Up: .6[-1+.9*0] + .2[0+.9*0] + .2[0+.9*0] = -.6
2. Right: .6[0+.9*0] + .2[-1+.9*0] + .2[0+.9*0] = -.2
3. Down: .6[0+.9*0] + .2[0+.9*0] + .2[0+.9*0] = 0
4. Left: .6[0+.9*0] + .2[0+.9*0] + .2[-1+.9*0] = -2

   New Value : 0

All other states have no updates since neighbors initial state is 0 and no neighbors have a reward and thus stay 0

Iteration 2:

3,3:

1. Up: .6[0+.9*.6] + .2[0+.9*0] + .2[1+.9*0] = .524
2. Right: .6[1+.9*0] + .2[0+.9*.6] + .2[0+.9*0] = .708
3. Down: .6[0+.9*0] + .2[0+.9*0] + .2[1+.9*0] = .2
4. Left: .6[0+.9*0] + .2[0+.9*0] + .2[0+.9*.6] = .108

   New Value : .708

2,3:

1. Up: .6[0+.9*0] + .2[0+.9*0] + .2[0+.9*.6] = .108
2. Right: .6[0+.9*.6] + .2[0+.9*0] + .2[0+.9*0] = .324
3. Down: .6[0+.9*0] + .2[0+.9*.6] + .2[0+.9*0] = .108
4. Left: .6[0+.9*0] + .2[0+.9*0] + .2[0+.9*0] = 0

   New Value : .324

3,2:

1. Up: .6[0+.9*.6] + .2[0+.9*0] + .2[-1+.9*0] = .124
2. Right: .6[-1+.9*0] + .2[0+.9*.6] + .2[0+.9*0] = -.492
3. Down: .6[0+.9*0] + .2[-1+.9*0] + .2[0+.9*0] = -.2
4. Left: .6[0+.9*0] + .2[0+.9*0] + .2[0+.9*.6] = .108

   New Value : .124

All other states have no updates since neighbors and selves had no updates and their previous state was 0.

Problem 2:

a) Iteration 1:

   Quit: 1*[5+0] = 5

   Continue: .33*[5+0] + .66[3+0] = 3.63

   V(In) = 5

b) Iteration 2:

   Quit: 1*[5+0] = 5

   Continue: .33*[5+0] + .66[3+5] = 6.93

   V(In) = 6.93

c) Iteration 3:

   Quit: 1*[5+0] = 5

   Continue: .33*[5+0] + .66[3+6.93] = 8.2

   V(In) = 8.2

d) Policy Discussion:

   The agent might have a preference for immediate reward meaning it would prefer the immediate 5 points over the expected 3.63 in the first iteration.

Problem 3:

Iteration 1:

Q Table:

3,3:

1. Up: .8[0] + .1[0] + .1[1] = .1
2. Right: .8[1] + .1[0] + .1[0] = .8
3. Down: .8[0] + .1[1] + .1[0] = .1
4. Left: .8[0] + .1[0] + .1[0] = 0

3,2:

1. Up: .8[0] + .1[0] + .1[-1] = -.1
2. Right: .8[-1] + .1[0] + .1[0] = -.8
3. Down: .8[0] + .1[-1] + .1[0] = -.1
4. Left: .8[0] + .1[0] + .1[0] = 0

4,1:

1. Up: .8[-1] + .1[0] + .1[0] = -.8
2. Right: .8[0] + .1[-1] + .1[0] = -.1
3. Down: .8[0] + .1[0] + .1[0] = 0

4. Left: .8[0] + .1[0] + .1[-1] = -.1

All other states have no updates since neighbors initial state is 0 and no neighbors have a reward and thus stay 0 or are terminal

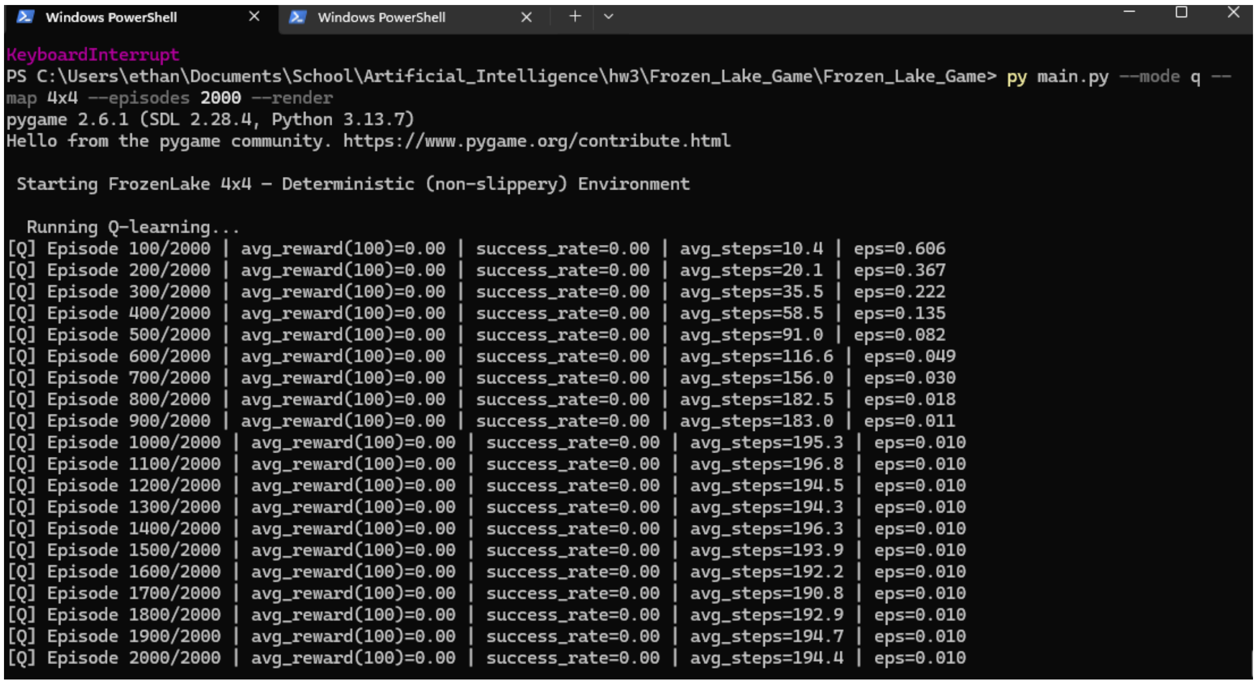
V Table:

3,3: max a of Q((3,3),a) = .8
3,2: max a of Q((3,2),a) = 0
4,1: max a of Q((4,1),a) = 0

All other states have all 0 Q values or are terminal

:

Problem 4:

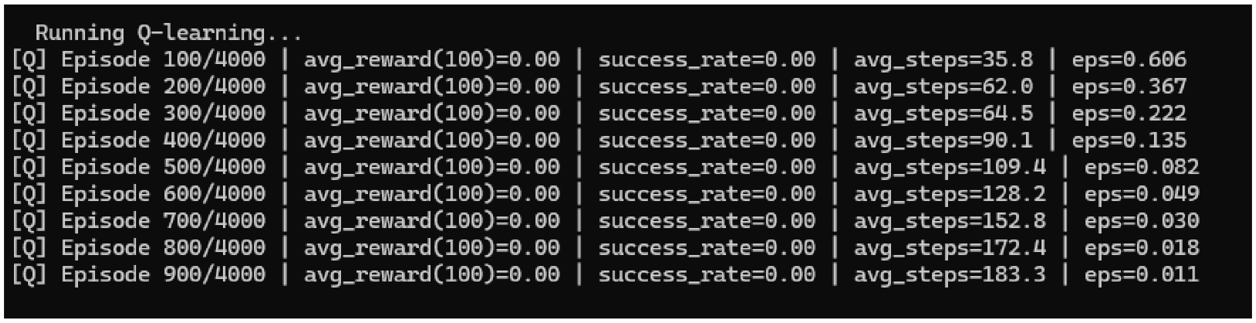

Windows PowerShell | Windows PowerShell

```
KeyboardInterrupt
PS C:\Users\ethan\Documents\School\Artificial_Intelligence\hw3\Frozen_Lake_Game\Frozen_Lake_Game> py main.py --mode q --
map 4x4 --episodes 2000 --render
pygame 2.6.1 (SDL 2.28.4, Python 3.13.7)
Hello from the pygame community. https://www.pygame.org/contribute.html

 Starting FrozenLake 4x4 - Deterministic (non-slippery) Environment

  Running Q-learning...
[Q] Episode 100/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=10.4 | eps=0.606
[Q] Episode 200/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=20.1 | eps=0.367
[Q] Episode 300/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=35.5 | eps=0.222
[Q] Episode 400/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=58.5 | eps=0.135
[Q] Episode 500/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=91.0 | eps=0.082
[Q] Episode 600/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=116.6 | eps=0.049
[Q] Episode 700/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=156.0 | eps=0.030
[Q] Episode 800/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=182.5 | eps=0.018
[Q] Episode 900/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=183.0 | eps=0.011
[Q] Episode 1000/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=195.3 | eps=0.010
[Q] Episode 1100/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=196.8 | eps=0.010
[Q] Episode 1200/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=194.5 | eps=0.010
[Q] Episode 1300/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=194.3 | eps=0.010
[Q] Episode 1400/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=196.3 | eps=0.010
[Q] Episode 1500/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=193.9 | eps=0.010
[Q] Episode 1600/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=192.2 | eps=0.010
[Q] Episode 1700/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=190.8 | eps=0.010
[Q] Episode 1800/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=192.9 | eps=0.010
[Q] Episode 1900/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=194.7 | eps=0.010
[Q] Episode 2000/2000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=194.4 | eps=0.010
```

I) The policy did not evolve except for epsilon making the agent prefer going left since the epsilon decayed too fast for exploration. If the decay is adjusted the agent evolves to select the right path very optimally. In addition if a negative reward is introduced for falling into the ice pit then it also reaches the path even with bad decay.
II) Very suboptimal, optimal with proper epsilon decay or negative reward
III) The agent prefers to go to the left and does not explore much since there is no randomness. With better epsilon decay or negative reward the agent finds the optimal path very quickly and follows it consistently since there is little randomness



```
  Running Q-learning...
[Q] Episode 100/4000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=35.8 | eps=0.606
[Q] Episode 200/4000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=62.0 | eps=0.367
[Q] Episode 300/4000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=64.5 | eps=0.222
[Q] Episode 400/4000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=90.1 | eps=0.135
[Q] Episode 500/4000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=109.4 | eps=0.082
[Q] Episode 600/4000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=128.2 | eps=0.049
[Q] Episode 700/4000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=152.8 | eps=0.030
[Q] Episode 800/4000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=172.4 | eps=0.018
[Q] Episode 900/4000 | avg_reward(100)=0.00 | success_rate=0.00 | avg_steps=183.3 | eps=0.011
```

I) The policy did not evolve except for epsilon making the agent prefer going left since the epsilon decayed too fast for exploration.
II) Very suboptimal
III) The agent prefers to go to the left but explores more due to the slipperyness. Still this was not enough resulting in the agent not figuring out anything. I did not run the full 4000 iterations because the steps took much too long due to the q table still being 0 it just wanted to go into the wall wasting steps and taking more time even at max framerate. With negative reward alone this agent still does not perform well due to fast epsilon decay. Perhaps with better epsilon decay and negative rewards it will be able to find the best policy.

Note: I submitted two copies of my code, one with the normal reward and one with negative reward for hitting the holes in the ice.