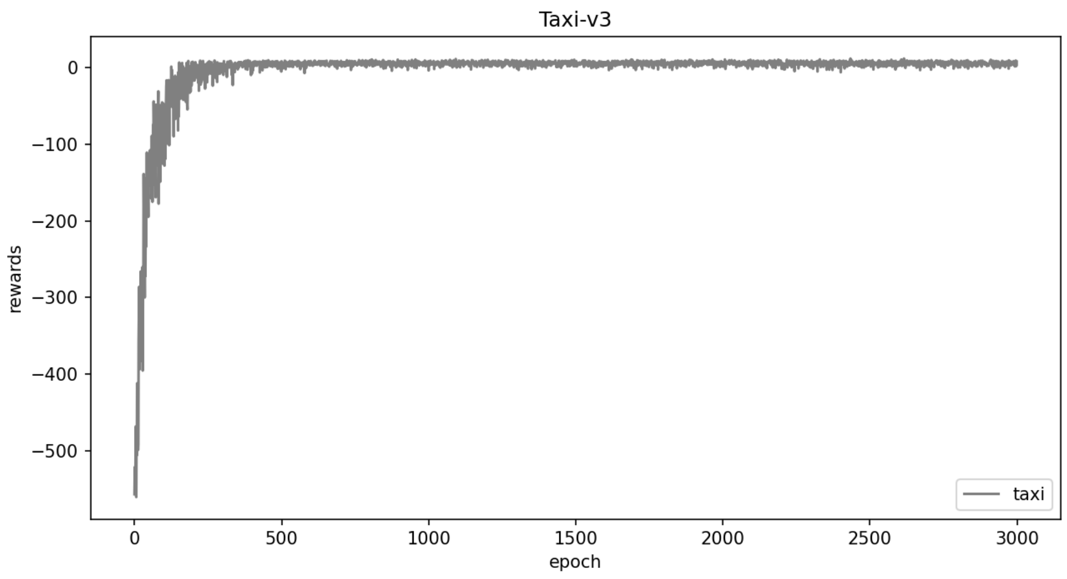
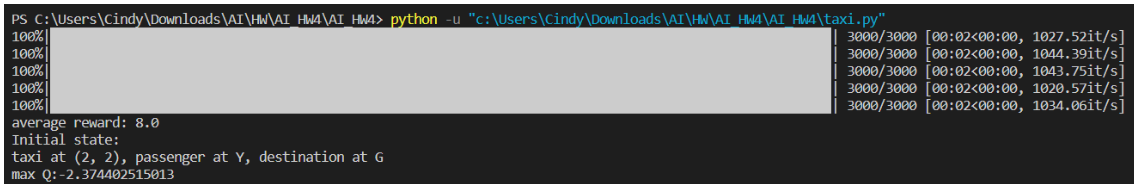
Homework 4:
Reinforcement Learning
Report Template

Part I. Experiment Results (the score here is included in your implementation):

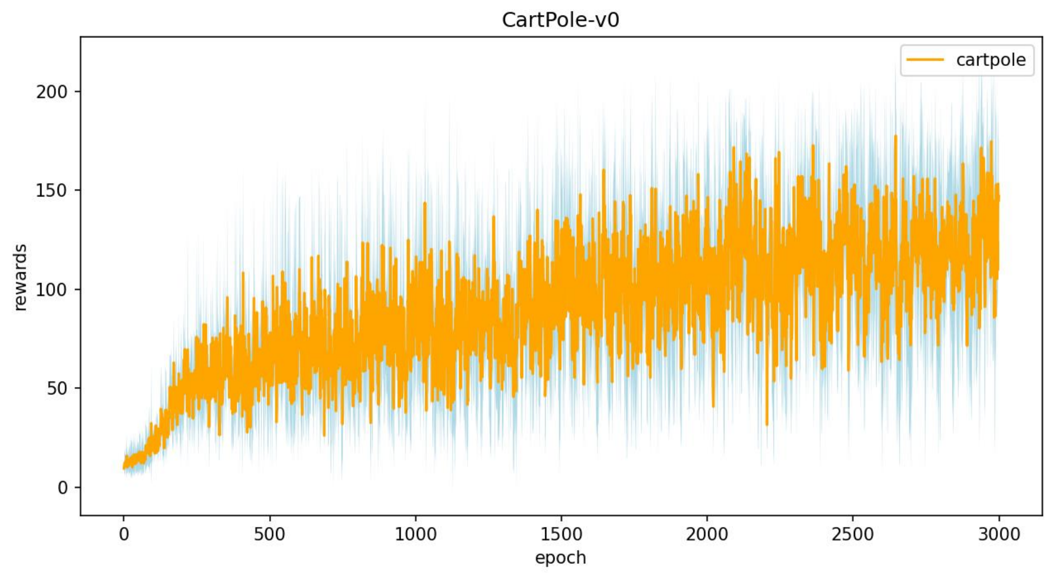
1. taxi.png:

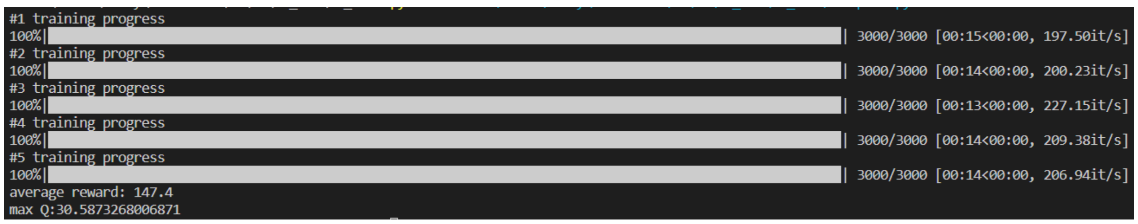




```
PS C:\Users\Cindy\Downloads\AI\HW\AI_HW4\AI_HW4> python -u "c:\Users\Cindy\Downloads\AI\HW\AI_HW4\AI_HW4\taxi.py"
100%|████████████████████████████████████████████████████████████████| 3000/3000 [00:02<00:00, 1027.52it/s]
100%|████████████████████████████████████████████████████████████████| 3000/3000 [00:02<00:00, 1044.39it/s]
100%|████████████████████████████████████████████████████████████████| 3000/3000 [00:02<00:00, 1043.75it/s]
100%|████████████████████████████████████████████████████████████████| 3000/3000 [00:02<00:00, 1020.57it/s]
100%|████████████████████████████████████████████████████████████████| 3000/3000 [00:02<00:00, 1034.06it/s]
average reward: 8.0
Initial state:
taxi at (2, 2), passenger at Y, destination at G
max Q:-2.374402515013
```

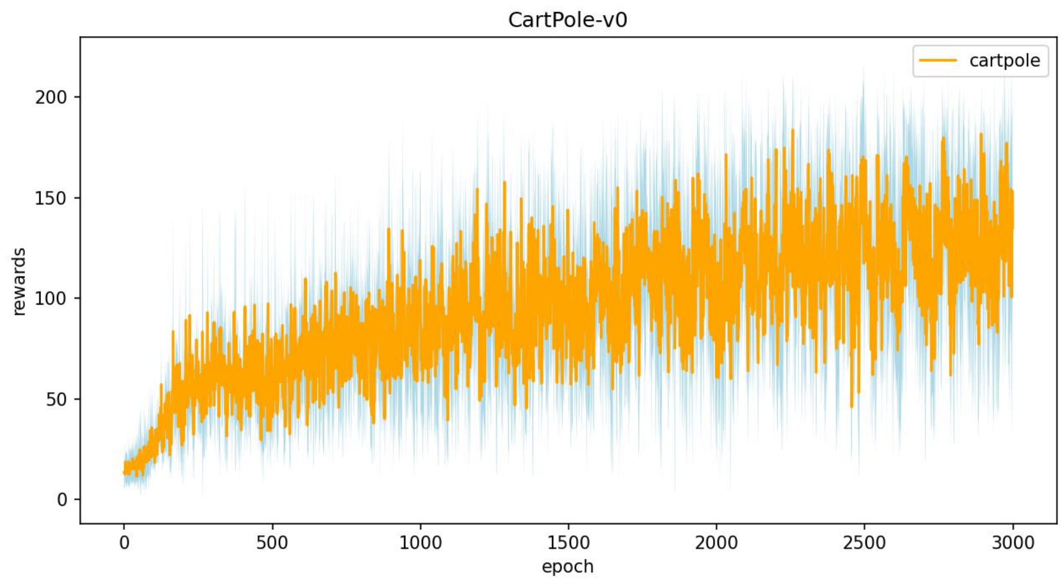
2. cartpole.png:

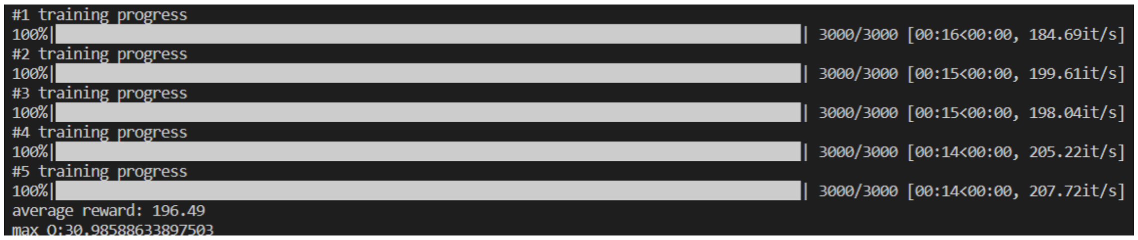




```
#1 training progress
100%|██████████████████████████████████████████████████| 3000/3000 [00:15<00:00, 197.50it/s]
#2 training progress
100%|██████████████████████████████████████████████████| 3000/3000 [00:14<00:00, 200.23it/s]
#3 training progress
100%|██████████████████████████████████████████████████| 3000/3000 [00:13<00:00, 227.15it/s]
#4 training progress
100%|██████████████████████████████████████████████████| 3000/3000 [00:14<00:00, 209.38it/s]
#5 training progress
100%|██████████████████████████████████████████████████| 3000/3000 [00:14<00:00, 206.94it/s]
average reward: 147.4
max Q:30.5873268006871
```

The result above is what I get using my own seed, which is 118.

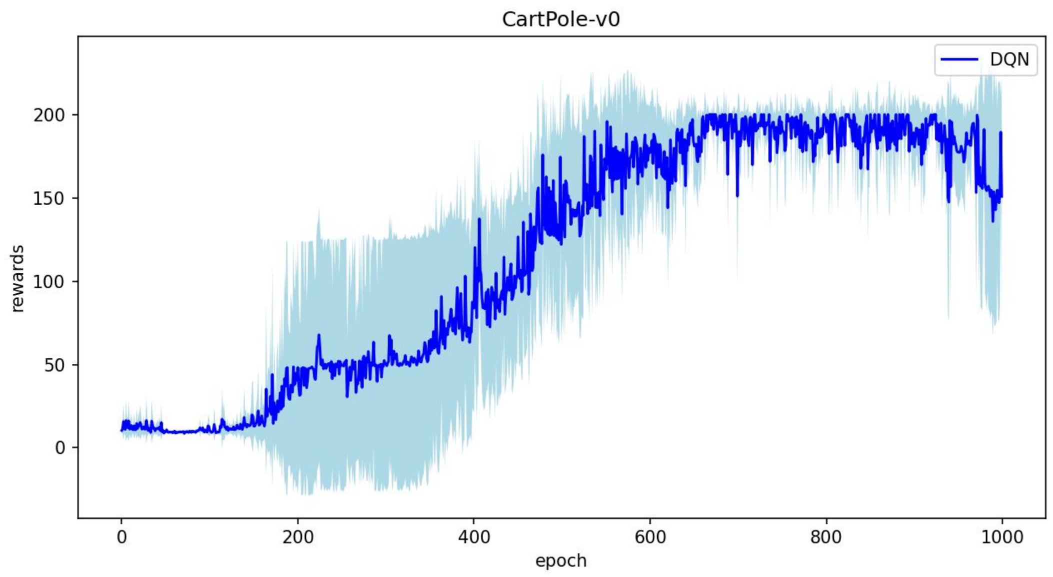


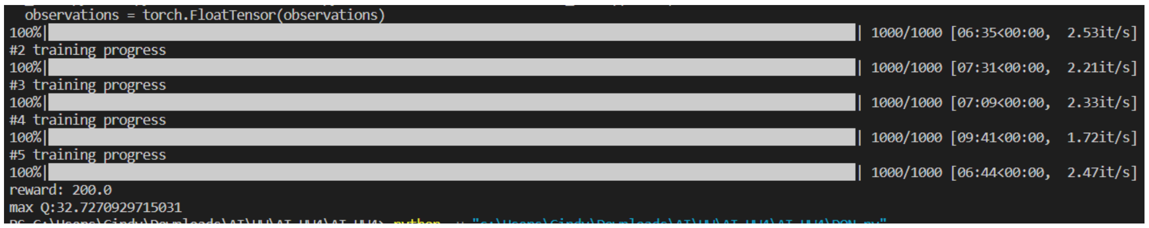


```
#1 training progress
100%|██████████████████████████████████████████████| 3000/3000 [00:16<00:00, 184.69it/s]
#2 training progress
100%|██████████████████████████████████████████████| 3000/3000 [00:15<00:00, 199.61it/s]
#3 training progress
100%|██████████████████████████████████████████████| 3000/3000 [00:15<00:00, 198.04it/s]
#4 training progress
100%|██████████████████████████████████████████████| 3000/3000 [00:14<00:00, 205.22it/s]
#5 training progress
100%|██████████████████████████████████████████████| 3000/3000 [00:14<00:00, 207.72it/s]
average reward: 196.49
max Q:30.98588633897503
```

This is the result I get using other seed, which is 100. The average reward is way much higher than that of my seed.

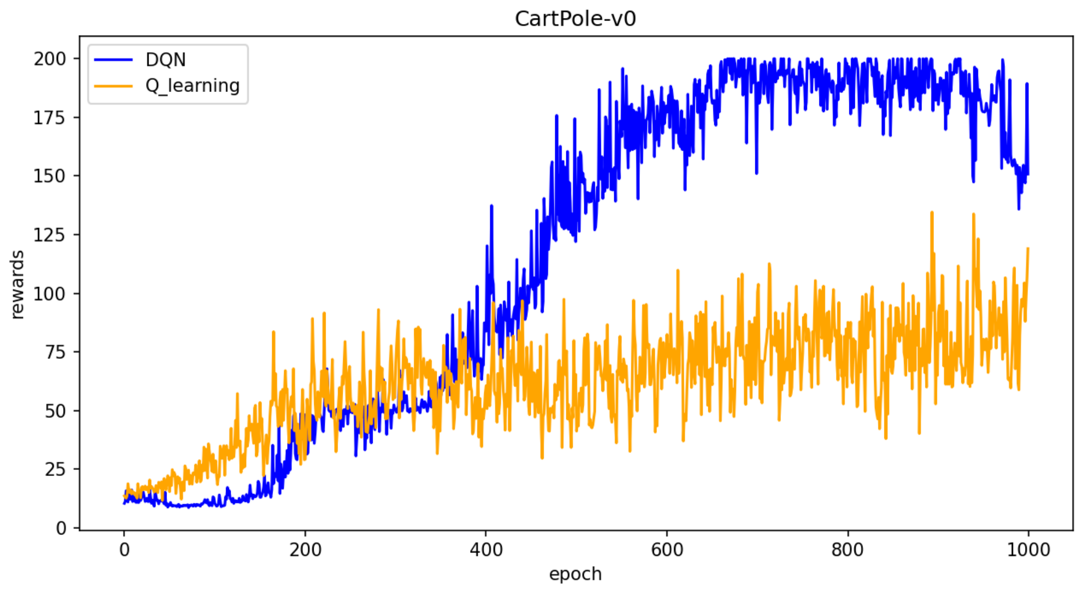
3. DQN.png:





```
  observations = torch.FloatTensor(observations)
100%|██████████████████████████████████████████████████| 1000/1000 [06:35<00:00,  2.53it/s]
#2 training progress
100%|██████████████████████████████████████████████████| 1000/1000 [07:31<00:00,  2.21it/s]
#3 training progress
100%|██████████████████████████████████████████████████| 1000/1000 [07:09<00:00,  2.33it/s]
#4 training progress
100%|██████████████████████████████████████████████████| 1000/1000 [09:41<00:00,  1.72it/s]
#5 training progress
100%|██████████████████████████████████████████████████| 1000/1000 [06:44<00:00,  2.47it/s]
reward: 200.0
max Q:32.7270929715031
```

4. compare.png:

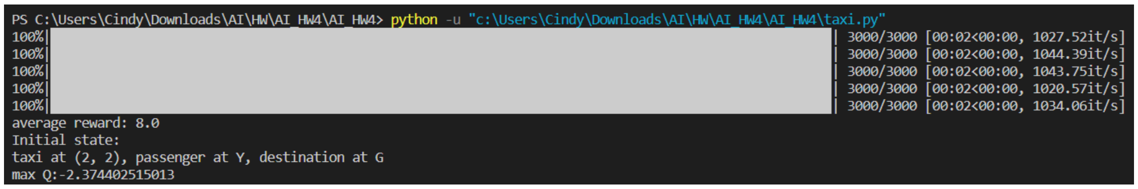


Part II. Question Answering (50%):

1. Calculate the optimal Q-value of a given state in Taxi-v3 (the state is assigned in google sheet), and compare with the Q-value you learned (Please screenshot the result of the “check_max_Q” function to show the Q-value you learned). **(4%)**

$$(-1) + \left[(-1) \times \left((0.9)^1 + (0.9)^2 + (0.9)^3 + \ldots + (0.9)^{10} \right) + 20 \times (0.9)^{11} \right.$$

$$= (-1) + (-1) \times \left(\frac{(0.9)^{11} - (0.9)}{0.1} \right) + 20 \times (0.9)^{11} \approx -0.586$$



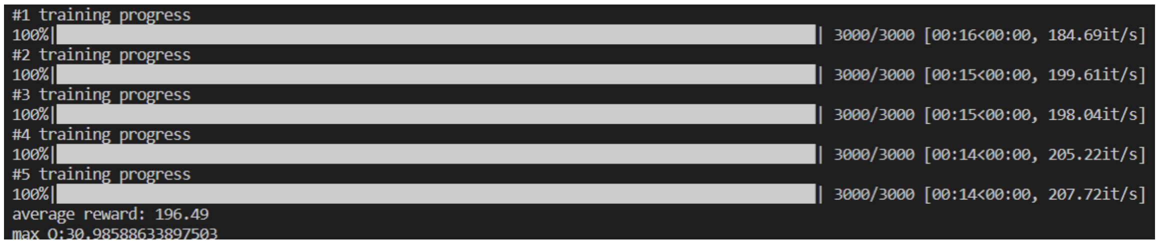
```
PS C:\Users\Cindy\Downloads\AI\HW\AI_HW4\AI_HW4> python -u "c:\Users\Cindy\Downloads\AI\HW\AI_HW4\AI_HW4\taxi.py"
100%|██████████████████████████████████████████████████| 3000/3000 [00:02<00:00, 1027.52it/s]
100%|██████████████████████████████████████████████████| 3000/3000 [00:02<00:00, 1044.39it/s]
100%|██████████████████████████████████████████████████| 3000/3000 [00:02<00:00, 1043.75it/s]
100%|██████████████████████████████████████████████████| 3000/3000 [00:02<00:00, 1020.57it/s]
100%|██████████████████████████████████████████████████| 3000/3000 [00:02<00:00, 1034.06it/s]
average reward: 8.0
Initial state:
taxi at (2, 2), passenger at Y, destination at G
max Q:-2.374402515013
```

From the calculation and the optimal Q-value that I got from Taxi-v3, it is slightly different that the max Q in the program is lower.

2. Calculate the max Q-value of the initial state in CartPole-v0, and compare with the Q-value you learned. (Please screenshot the result of the “check_max_Q” function to show the Q-value you learned) **(4%)**

$$1 + (0.97) + (0.97)^2 + (0.97)^3 + \ldots + (0.97)^{199}$$

$$= \frac{1 - (0.97)^{200}}{0.03} \approx 33.258$$



```
#1 training progress
100%|██████████████████████████████████████████████████| 3000/3000 [00:16<00:00, 184.69it/s]
#2 training progress
100%|██████████████████████████████████████████████████| 3000/3000 [00:15<00:00, 199.61it/s]
#3 training progress
100%|██████████████████████████████████████████████████| 3000/3000 [00:15<00:00, 198.04it/s]
#4 training progress
100%|██████████████████████████████████████████████████| 3000/3000 [00:14<00:00, 205.22it/s]
#5 training progress
100%|██████████████████████████████████████████████████| 3000/3000 [00:14<00:00, 207.72it/s]
average reward: 196.49
max Q:30.98588633897503
```

The optimal value gained is again lower in the cartpole program too.

3.
   a. Why do we need to discretize the observation in Part 2? **(2%)**

- Because in the cartpole.py, the state space is continuous. Thus, we need to discretize the states before building the Q-table.

b. How do you expect the performance will be if we increase "num_bins"? **(2%)**
- The performance will be better

c. Is there any concern if we increase "num_bins"? **(2%)**
- As we increase num_bins, we might need more space in Q-table to store the states.

4. Which model (DQN, discretized Q learning) performs better in Cartpole-v0, and what are the reasons? **(3%)**
    - DQN performs better than discretized Q learning as we can see in the average reward that we get in the programs. DQN can get Q-value for more states compared to discretized Q learning.

5.
    a. What is the purpose of using the epsilon greedy algorithm while choosing an action? **(2%)**
    - The epsilon helps balance between exploration and exploitation.

    b. What will happen, if we don't use the epsilon greedy algorithm in the CartPole-v0 environment? **(3%)**
    - The result might be more inefficient as there is no balance between exploration and exploitation, and it might just take random action.

    c. Is it possible to achieve the same performance without the epsilon greedy algorithm in the CartPole-v0 environment? Why or Why not? **(3%)**
    - Yes, it is possible to achieve the same performance using other algorithm or maybe applying some conditions to balance between exploration and exploitation. There might be an algorithm called Boltzmann algorithm that might suits this.

    d. Why don't we need the epsilon greedy algorithm during the testing section? **(2%)**
    - Because we have saved the Q-values in the Q-table during training section, and thus only need to choose the best action according to the Q-table.

6. Why is there "`with torch.no_grad():`" in the "choose_action" function in DQN? **(3%)**
    - It works by setting requires_grad as false, which can reduce memory consumption.

7.
    a. Is it necessary to have two networks when implementing DQN? **(1%)**

        - It is not necessary.

    b. What are the advantages of having two networks? **(3%)**
        - It can balance the algorithm by having two separate networks.

    c. What are the disadvantages? **(2%)**
        - The disadvantage of having two networks is that it’s trained slower than that of one network.

8.
    a. What is a replay buffer(memory)? Is it necessary to implement a replay buffer? What are the advantages of implementing a replay buffer? **(5%)**
        - It is to store the previous tuples or the history and will be added to buffer as we explore. Thus, it is necessary to implement it. The advantage is that it will has better result, as it will choose the best action that produce maximum rewards.

    b. Why do we need batch size? **(3%)**
        - Batch size is the number of trainings utilized in one iteration and will make the program faster.

    c. Is there any effect if we adjust the size of the replay buffer(memory) or batch size? Please list some advantages and disadvantages. **(2%)**
        - Yes, smaller batch size and smaller replay buffer means faster training. However, the disadvantage of them respectively are less accuracy and unstable.

9.
    a. What is the condition that you save your neural network? **(1%)**
        - Target network updated every 100 times

    b. What are the reasons? **(2%)**
        - Increase accuracy and faster training because not saving it every iteration.

10. What have you learned in the homework? **(2%)**
    - I have learned a lot from this homework, including the implementation of Q-learning and DQN. Unlike what we only learn during the class of Q-values and Q-tables, it feels amazing and confusing as well when I try to implement it. It has bring me lots of challenges as well as difficulties, but on the same time, it has also taught me a lot of things.