Problem 1 (Q-Value Iteration) (15+15=30 points)

(a) Recall that in Lecture 3, we define $V^*(s) := \max_\pi V^\pi(s)$ and $Q^*(s,a) := \max_\pi Q^\pi(s,a)$. Suppose $\gamma \in (0,1)$. Prove the following Bellman optimality equations:

$$V^*(s) = \max_a Q^*(s,a) \tag{1}$$

$$Q^*(s,a) = R_{s,a} + \gamma \sum_{s'} P^a_{ss'} V^*(s'). \tag{2}$$

Please carefully justify every step of your proof. (Hint: For (1), you may first prove that $V^*(s) \le \max_a Q^*(s,a)$ and then show $V^*(s) < \max_a Q^*(s,a)$ cannot happen by contradiction. On the other hand, (2) can be shown by using the similar argument or by leveraging the fact that $Q^\pi(s,a) = R_{s,a} + \gamma \sum_{s'} P^a_{ss'} V^\pi(s')$)

P1 (a)

Show $V^*(s) \le \max_a Q^*(s,a)$:

Definition :

$$V^*(s) = \max_{\pi \in \Pi} V^\pi(s)$$

$$Q^*(s,a) = \max_{\pi \in \Pi} Q^\pi(s,a)$$

$$V^\pi(s) = \sum_{a \in A} \pi(s|a) \cdot Q^\pi(s,a)$$

$$Q^\pi(s,a) = R_{s,a} + \sum_{s' \in S} P^a_{ss'} \cdot V^\pi(s')$$

$$V^*(s) = \max_{\pi \in \Pi} V^\pi(s)$$

$$= \max_{\pi \in \Pi} \left\{ \sum_{a \in A} \left[\pi(s|a) \cdot Q^\pi(s,a) \right] \right\}$$

$$\le \max_{\pi \in \Pi} \left\{ \sum_{a \in A} \left[\pi(s|a) \cdot \max_{a'} \left(Q^\pi(s,a') \right) \right] \right\}$$

$$\le \max_{\pi \in \Pi} \left\{ \sum_{a \in A} \left[\pi(s|a) \right] \right\} \cdot \max_{a'} \left\{ Q^\pi(s,a') \right\}$$

$$= \max_{a'} Q^\pi(s,a')$$

$$= \max_a Q^\pi(s,a)$$

Show $V^*(s) < \max_a Q^*(s,a)$ cannot happen by contradiction :

Suppose $V^*(s) < \max_a Q^*(s,a)$

Then there exist a policy π' (first action $= \arg\max_a Q^*(s,a)$, then follow the policy $V^*(s)$ goes.) such that

$$V^{\pi'} = \sum_{a \in A} \pi(a|s) \cdot Q^*(s,a) > V^*(s)$$

However, by defintion of $V^*(s) = \max_{\pi \in \Pi} V^\pi(s)$

$\Rightarrow V^*(s) \geq V^{\pi'}(s) \Rightarrow$ contradiction

So $V^*(s) = \max_a Q^*(s,a)$

Show $Q^*(s,a) = R_{s,a} + \gamma \sum_{s'} P^a_{ss'} V^*(s')$

$$\begin{aligned} Q^*(s,a) &= \max_{\pi \in \Pi} Q^\pi(s,a) \\ &= \max_{\pi \in \Pi} \left\{ R_{s,a} + \sum_{s'} P^a_{ss'} \cdot V^\pi(s') \right\} \\ &= R_{s,a} + \sum_{s'} \left[P^a_{ss'} \cdot \max_{\pi \in \Pi} V^\pi(s') \right] \} \\ &= R_{s,a} + \sum_{s'} P^a_{ss'} \cdot V^*(s) \end{aligned}$$

(b) Based on (a), we thereby have the recursive Bellman optimality equation for the optimal action-value function Q_* as:

$$Q^*(s,a) = R_{s,a} + \gamma \sum_{s'} P^a_{ss'} \big(\max_{a'} Q^*(s',a') \big) \tag{3}$$

Similar to the standard Value Iteration, we can also study the *Q-Value Iteration* by defining the Bellman optimality operator $T^* : \mathbb{R}^{|\mathcal{S}||\mathcal{A}|} \to \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ for the action-value function: for every state-action pair (s,a)

$$[T^*(Q)](s,a) := R_{s,a} + \gamma \sum_{s'} P^a_{ss'} \max_{a'} Q(s',a') \tag{4}$$

Show that the operator T^* is a γ-contraction operator in terms of ∞-norm. Please carefully justify every step of your proof. (Hint: For any two action-value functions Q, Q', we have $\|T^*(Q) - T^*(Q')\|_\infty = \max_{(s,a)} |[T^*(Q)](s,a) - [T^*(Q')](s,a)|$)

P1 (b)

Show T^* is a γ-contraction operator in terms of ∞-norm :

$$\|T^*(Q) - T^*(Q')\|_\infty = \max_{(s,a)} \left| [T^*(Q)](s,a) - [T^*(Q')](s,a) \right|$$

$$= \max_{(s,a)} \left| \left[R_{s,a} + \gamma \sum_{s'} P^a_{ss'} \max_{a'} Q(s',a') \right] - \left[R_{s,a} + \gamma \sum_{s'} P^a_{ss'} \max_{a'} Q'(s',a') \right] \right|$$

$$= \max_{(s,a)} \left| \gamma \sum_{s'} P^a_{ss'} \max_{a'} Q(s',a') - \gamma \sum_{s'} P^a_{ss'} \max_{a'} Q'(s',a') \right|$$

$$= \max_{(s,a)} \left| \gamma \sum_{s'} P^a_{ss'} \left[\max_{a'} Q(s',a') - \max_{a'} Q'(s',a') \right] \right|$$

$$\le \max_{(s,a)} \left\{ \gamma \sum_{s'} P^a_{ss'} \left| \max_{a'} Q(s',a') - \max_{a'} Q'(s',a') \right| \right\}$$

$$\le \max_{(s,a)} \left\{ \gamma \sum_{s'} P^a_{ss'} \max_{(s'',a'')} \left| Q(s'',a'') - Q'(s'',a'') \right| \right\}$$

$$= \max_{(s,a)} \left\{ \gamma \sum_{s'} P^a_{ss'} \|Q - Q'\|_\infty \right\}$$

$$= \gamma \|Q - Q'\|_\infty$$

($\sum_{s'} P^a_{ss'} = 1$)

$\Rightarrow T^*$ is a γ-contraction operator

Problem 2 (Regularized MDPs) (10+10=20 points)

In Lecture 4, we formally describe the regularized MDP, which is a direct extension of the classic MDP with a regularizer Ω. In this problem, for simplicity, suppose we use the Shannon entropy as our regularizer, i.e., $\Omega(\pi(\cdot|s)) \equiv H(\pi(\cdot|s)) := -\sum_{a\in\mathcal{A}} \pi(a|s) \ln \pi(a|s)$. Let us verify a few important properties mentioned in Lecture 4 as follows.

(a) Recall that we introduce the "regularized Bellman expectation operator" T^π_Ω as

$$[T^\pi_\Omega V](s) := R^\pi_s + \Omega(\pi(\cdot|s)) + \gamma P^\pi_{ss'} V. \tag{5}$$

Please verify that T^π_Ω is a contraction operator in L_∞ norm. (Hint: Try to extend the proof procedure of the contraction property of T^π in Lecture 3)

P2 (a)

Show T^π_Ω is a γ-contraction operator

$$\|T^*(V) - T^*(V')\|_\infty = \max_{s\in S} \left| [T^\pi_\Omega V](s) - T^\pi_\Omega V'](s) \right|$$

$$= \max_{s\in S} \left| \left[R^\pi_s + \Omega(\pi(\cdot|s)) + \gamma P^\pi_{ss'} V(s')\right] - \left[R^\pi_s + \Omega(\pi(\cdot|s)) + \gamma P^\pi_{ss'} V'(s')\right] \right|$$

$$= \max_{s\in S} \left| \left[R^\pi_s - \sum_{a\in A} \pi(a|s) \ln(\pi(a|s)) + \gamma P^\pi_{ss'} V(s')\right] - \left[R^\pi_s - \sum_{a\in A} \pi(a|s) \ln(\pi(a|s)) + \gamma P^\pi_{ss'} V'(s')\right] \right|$$

$$= \max_{s\in S} \left| \gamma P^\pi_{ss'} V(s') - \gamma P^\pi_{ss'} V'(s') \right|$$

$$= \max_{s\in S} \left| \gamma P^\pi_{ss'} [V(s') - V'(s')] \right|$$

$$\le \max_{s\in S} \left| \gamma [V(s') - V'(s')] \right| \quad (P^\pi_{ss'} \le 1)$$

$$= \gamma \|V - V'\|_\infty$$

(b) Moreover, under regularized MDPs, we study the optimal value functions V_Ω^* and optimal Q functions Q_Ω^* and learn the Bellman optimality equations as

$$V_\Omega^*(s) = \max_{\pi \in \Pi} R_s^\pi + \gamma P_s^\pi V_\Omega^* \tag{6}$$

$$Q_\Omega^*(s,a) = R_{s,a} + \gamma E_{s' \sim P(\cdot|s,a)}[V_\Omega^*(s')]. \tag{7}$$

Could you design an iterative algorithm that can obtain V_Ω^* and Q_Ω^*? Please clearly write down the complete pseudo code of your algorithm and provide comments on each line of your pseudo code. (Hint: Try to extend the Value Iteration for standard MDPs to the regularized MDPs based on Equation 6)

P2 (b)

Initialize $k \leftarrow 0$

For all state $s \in S$

Initialize value $V_{0,\Omega}(s) \leftarrow 0$ (set value of each state to 0)

While $V_{k,\Omega}$ haven't converge to V_Ω^* ($V_{k+1,\Omega} != V_{k,\Omega}$)

$V_{k+1,\Omega} \leftarrow \max_{\pi \in \Pi} (R_s^\pi + \gamma P_{ss'}^\pi V_{k,\Omega})$ ($V_{k+1} \leftarrow T_\Omega^\pi(V_k)$)

$k \leftarrow k+1$ (increment)

For all action $a \in A$

For all state $s \in S$

$Q_\Omega^*(s,a) = R_{s,a} + \gamma E_{s' \sim P(\cdot|s,a)}[V_{k,\Omega}(s')]$ (現在 $V_{k,\Omega} = V_\Omega^*$, 計算每一個 $Q_\Omega^*(s,a)$)

Return $V_\Omega^*(s)$, $Q_\Omega^*(s,a)$

Problem 3 (A Property Used in Policy Gradient) (10 points)

Show the following useful property discussed in Lectures 5-6: for any function $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$,

$$\mathbb{E}_{\tau \sim P_\mu^{\pi_\theta}}\left[\sum_{t=0}^{\infty} \gamma^t f(s_t, a_t)\right] = \frac{1}{1-\gamma}\mathbb{E}_{s \sim d_\mu^{\pi_\theta}}\mathbb{E}_{a \sim \pi_\theta(\cdot|s)}\left[f(s,a)\right] \tag{8}$$

(Hint: It might be slightly easier to go from the RHS to LHS. Specifically, you may first expand the RHS of (8) into a sum of $f(s,a)$ over s and a and then apply the definition of $d_\mu^{\pi_\theta}$, which involves a sum of probability over t. Next, try to reorganize the triple summation into the form of the LHS of (8))

P3

$$\frac{1}{1-\gamma} E_{s\sim d_\mu^{\pi_\theta}} E_{a\sim\pi_\theta(\cdot|s)}[f(s,a)] = \frac{1}{1-\gamma}\sum_{s\in S}\sum_{a\in A} d_\mu^{\pi}(s)\cdot\pi_\theta(a|s)\cdot f(s,a)$$

$$= \frac{1}{1-\gamma}\sum_{s\in S}\sum_{a\in A}\sum_{s_0\in\mu}\mu(s_0) d_{s_0}^{\pi_\theta}(s)\, d_\mu^{\pi}(s)\cdot\pi_\theta(a|s)\cdot f(s,a)$$

$$= \frac{1}{1-\gamma}\sum_{s\in S}\sum_{a\in A}\sum_{s_0\in\mu}(1-\gamma)\sum_{t=0}^{\infty}\gamma^t P(s_t=s|s_0,\pi_\theta)\cdot\mu(s_0)\cdot\pi_\theta(a|s)\cdot f(s,a)$$

$$= \sum_{s_0\in\mu}\sum_{s\in S}\sum_{a\in A}\left[\sum_{t=0}^{\infty}\gamma^t\cdot f(s,a)\right]\mu(s_0)\cdot P(s_t=s|s_0,\pi_\theta)\cdot\pi_\theta(a|s)$$

τ (trajectory)

$$= E_{\tau\sim P_\mu^{\pi_\theta}}\left[\sum_{t=0}^{\infty}\gamma^t\cdot f(s_t,a_t)\right]$$

Problem 4 (Implementing Policy Iteration and Value Iteration) (35 points)

In this problem, we will implement Policy Iteration and Calue Iteration for a classic MDP environment called "Taxi" (Dietterich, 2000). This environment has been included in the OpenAI Gym: https://gym.openai.com/envs/Taxi-v3/. To accomplish this task, you may take the following steps:

No question

Problem 5 (Installing and Getting Familiar With D4RL Dataset) (10 points)

In this problem, you will be asked to install and investigate the D4RL Dataset, which is currently the standard dataset for Offline RL. To accomplish this task, you shall take the following steps:

- Please first take a careful look at the GitHub repo of D4RL https://github.com/Farama-Foundation/D4RL and the paper https://arxiv.org/abs/2004.07219. Then, install **d4rl** by following the installation guide on the GitHub repo (you may need to install several other packages, such as Gymnasium and MuJoCo).
- Read and run the provided **d4rl_sanity_check.py** and finish the following tasks:
  - (1) Generate an offline dataset by directly running **d4rl_sanity_check.py**.
  - (2) Describe the format of the dataset that you just obtained.
  - (3) Modify the **d4rl_sanity_check.py** and generate another offline dataset of one of the MuJoCo tasks (e.g., Hopper, Walker, or Halfcheetah). Similarly, describe the format of the MuJoCo dataset that you just obtained.
- One final remark: We will keep using the D4RL dataset for HW2, HW3, and the team final project. Therefore, it would be very helpful to now set up the environment that you could easily reuse subsequently.

(1) 我的執行結果如下圖，成功 generate dataset

```python
import gym
import d4rl # Import required to register environments, you may need to also import the submodulept

# Create the environmentexit
env = gym.make('maze2d-umaze-v1')

# d4rl abides by the OpenAI gym interface
env.reset()
env.step(env.action_space.sample())

# Each task is associated with a dataset
# dataset contains observations, actions, rewards, terminals, and infos
dataset = env.get_dataset()
print(dataset['observations']) # An N x dim_observation Numpy array of observations

# Alternatively, use d4rl.qlearning_dataset which
# also adds next_observations.
dataset = d4rl.qlearning_dataset(env)
```

```
/usr/local/lib/python3.10/dist-packages/gym/utils/seeding.py:38: DeprecationWarning: WARN: Function `rng.randn(*size)` is marked as deprecated and will be removed
  deprecation(
Downloading dataset: http://rail.eecs.berkeley.edu/datasets/offline_rl/maze2d/maze2d-umaze-sparse-v1.hdf5 to /root/.d4rl/datasets/maze2d-umaze-sparse-v1.hdf5
load datafile: 100%|██████████| 8/8 [00:00<00:00, 17.05it/s]
[[ 1.0856489   1.9745734   0.00981035  0.02174424]
 [ 1.0843927   1.97413    -0.12562364 -0.04433781]
 [ 1.0807577   1.9752754  -0.3634883   0.11453988]
 ...
 [ 1.1328583   2.8062387  -4.484303    0.09555068]
 [ 1.0883482   2.8068895  -4.4510083   0.06509537]
 [ 1.0463258   2.8074222  -4.202244    0.05324839]]
load datafile: 100%|██████████| 8/8 [00:00<00:00, 16.25it/s]
```

(2)我稍微修改了 code，以便觀察 dataset

```
import gym
import d4rl

env = gym.make('maze2d-umaze-v1')
env.reset()
dataset = env.get_dataset()

print("Observation Shape:", dataset['observations'].shape)
print("Action Shape:", dataset['actions'].shape)
print("Reward Shape:", dataset['rewards'].shape)
print("Terminal Shape:", dataset['terminals'].shape)

for _ in range(10):
        action = env.action_space.sample()
        print("Action at this time:", action)
        observation, reward, done, info = env.step(action)
        print("Observation:", observation)
        print("Reward:", reward)
        print("Done:", done)
        if done:
                observation = env.reset()
```

```
/usr/local/lib/python3.10/dist-packages/gym/utils/seeding.py:38: DeprecationWarning: WARN: Function `rng.randn(*size)` is mark
  deprecation(
load datafile: 100%|██████████| 8/8 [00:00<00:00, 19.68it/s]
Observation Shape: (1000000, 4)
Action Shape: (1000000, 2)
Reward Shape: (1000000,)
Terminal Shape: (1000000,)
Action at this time: [ 0.97649056 -0.43192947]
Observation: [ 2.97255189  2.05689009  0.41748031 -0.03716483]
Reward: 0.0
Done: False
Action at this time: [-0.02585862  0.80586714]
Observation: [2.97665516 2.05843861 0.41032743 0.1548521 ]
Reward: 0.0
Done: False
Action at this time: [ 0.4385132  -0.26457137]
Observation: [2.98179304 2.05935333 0.51378816 0.09147196]
Reward: 0.0
Done: False
Action at this time: [-0.73913854 -0.6023491 ]
Observation: [ 2.98515833  2.05883129  0.33652843 -0.05220367]
Reward: 0.0
Done: False
Action at this time: [ 0.8190923 -0.5181357]
Observation: [ 2.99046638  2.05707649  0.5308051  -0.17548053]
Reward: 0.0
Done: False
Action at this time: [ 0.79902446 -0.6023787 ]
Observation: [ 2.99766478  2.05389121  0.71983965 -0.31852743]
Reward: 0.0
Done: False
Action at this time: [ 0.05523174 -0.5566804 ]
Observation: [ 3.00497757  2.04938771  0.73127946 -0.45034995]
Reward: 0.0
Done: False
Action at this time: [0.5666699  0.07277343]
Observation: [ 3.01362255  2.04506826  0.86449809 -0.43194538]
Reward: 0.0
Done: False
Action at this time: [-0.03808868  0.54048914]
Observation: [ 3.02215623  2.04204634  0.85336782 -0.30219167]
Reward: 0.0
Done: False
Action at this time: [-0.1474886  -0.43219653]
Observation: [ 3.03031832  2.03800229  0.81620896 -0.40440555]
Reward: 0.0
Done: False
```

我先 print 出 Observation, Action, Reward, terminal 的大小，分別是 (1000000,4), (1000000,2), (1000000), (1000000)，可以看出，states 數應該是 1000000，且有兩個 action，Observation 有 4 個不同的 feature，但我不是很確定是什麼。

接下來我隨機 sample 一個 action，iteration 10 次並在每次 iteration 時 print 出當前的資訊。可以看到，observation 的值是有在變化的，但幅度不大。Reward 都是 0 是一個奇怪的現象。Done 始終為 0 是合理的，畢竟不太可能那麼快就完成迷宮。

(3) 我另外選了 walker dataset

```
[18] import gym
     import d4rl

     env = gym.make('walker2d-medium-v2')
     env.reset()
     dataset = env.get_dataset()

     print("Observation Shape:", dataset['observations'].shape)
     print("Action Shape:", dataset['actions'].shape)
     print("Reward Shape:", dataset['rewards'].shape)
     print("Terminal Shape:", dataset['terminals'].shape)

     for _ in range(10):
         action = env.action_space.sample()
         observation, reward, done, info = env.step(action)
         print("Observation:", observation)
         print("Reward:", reward)
         print("Done:", done)
         if done:
             observation = env.reset()
```

```
/usr/local/lib/python3.10/dist-packages/ipykernel/ipkernel.py:283: DeprecationWarning: `should_run_async` will not call `transfor
  and should_run_async(code)
/usr/local/lib/python3.10/dist-packages/gym/spaces/box.py:84: UserWarning: WARN: Box bound precision lowered by casting to float3
  logger.warn(f"Box bound precision lowered by casting to {self.dtype}")
load datafile: 100%|██████████| 21/21 [00:02<00:00,  8.36it/s]Observation Shape: (1000000, 17)
Action Shape: (1000000, 6)
Reward Shape: (1000000,)
Terminal Shape: (1000000,)
Observation: [ 1.25177247e+00 -4.28160193e-03  2.74072676e-04  5.32918924e-04
 -2.28354993e-03  8.82391497e-03 -2.62726620e-02  5.84716355e-02
 -2.16603934e-01 -2.41045400e-01 -9.48874777e-01 -9.55106662e-01
  1.18438474e-01 -1.43404824e-01  1.19920386e+00 -5.06244592e+00
  1.00000000e+01]
Reward: 0.9104789210841541
Done: False
Observation: [ 1.25025027e+00 -1.54644469e-02 -1.15631813e-02  2.06907229e-03
 -1.92118183e-02  1.49913457e-02 -6.61173853e-02  1.29818854e-01
 -3.72991574e-01 -1.41038830e-01 -1.80858268e+00 -1.93943576e+00
  2.32297572e-01 -4.07051755e+00  4.26394149e-01 -5.01719813e+00
  3.36059028e+00]
Reward: 0.7039204795589478
Done: False
Observation: [ 1.24913392e+00 -5.39413836e-02 -5.50808919e-02  4.38958637e-03
 -4.52825427e-02 -1.14432585e-02 -1.00628950e-01  1.13665164e-01
 -9.35324230e-01 -1.56449049e-01 -7.79231479e+00 -8.90667980e+00
  3.26695430e-01 -2.47038536e+00 -6.99328902e+00 -3.62213165e+00
 -7.38381036e+00]
Reward: 0.34407978541995055
Done: False
Observation: [ 1.24696202e+00 -1.28582562e-01 -1.38356692e-01  7.10539556e-03
 -1.13918110e-01 -6.91778384e-02 -1.54840295e-01  1.00883018e-01
 -1.48532418e+00 -4.13414312e-01 -1.00000000e+01 -1.00000000e+01
  3.45092139e-01 -1.00000000e+01 -7.46178646e+00 -9.88149668e+00
  3.98153431e+00]
Reward: -0.21465074962280498
Done: False
Observation: [ 1.24377934e+00 -2.12176359e-01 -2.27003243e-01  4.59484698e-03
 -2.33357659e-01 -1.34188067e-01 -2.11515330e-01  7.11851882e-02
 -1.35067551e+00 -3.72706956e-01 -1.00000000e+01 -1.00000000e+01
 -8.25464792e-01 -1.00000000e+01 -8.75781689e+00 -4.31109003e+00
 -1.00000000e+01]
Reward: -0.420317707465029
Done: False
Observation: [ 1.24009088e+00 -3.02763811e-01 -3.23485140e-01  3.51667903e-06
 -2.84649083e-01 -2.29236202e-01 -2.18238270e-01 -1.89397644e-02
 -1.32779320e+00 -5.65220159e-01 -1.00000000e+01 -1.00000000e+01
 -3.70707390e-01 -2.76701619e+00 -1.00000000e+01  2.58063325e+00
```

code 部分只有把我剛剛修改過的 code load 的 dataset 換成另 walker，因為我認為我前面改的已經透露足夠多的資訊。

可以觀察到，walker 的 observation 跟 maze-2d 不太一樣，它是一維 17 element 的 array，裡面存的內容我不是很清楚，但和 maze-2d 比起來，它存的內容少太多了。

Walker 的 action 有 6 種不同的。同樣 10 次 iteration，我們可以看到它有一個 reward 值，並且會隨著 iteration 中不同的 action 產生變動。值得一提，它的 reward 是一直在變小，可能是因為我的 action 是亂選的。

```
Reward: -0.21465074962280498
Done: False
Observation: [ 1.24377934e+00 -2.12176359e-01 -2.27003243e-01  4.59484698e-03
 -2.33357659e-01 -1.34188067e-01 -2.11515330e-01  7.11851882e-02
 -1.35067551e+00 -3.72706956e-01 -1.00000000e+01 -1.00000000e+01
 -8.25464792e-01 -1.00000000e+01 -8.75781689e+00 -4.31109003e+00
 -1.00000000e+01]
Reward: -0.420317707465029
Done: False
Observation: [ 1.24009088e+00 -3.02763811e-01 -3.23485140e-01  3.51667903e-06
 -2.84649083e-01 -2.29236202e-01 -2.18238270e-01 -1.89397644e-02
 -1.32779320e+00 -5.65220159e-01 -1.00000000e+01 -1.00000000e+01
 -3.70707390e-01 -2.76701619e+00 -1.00000000e+01  2.58063325e+00
 -1.00000000e+01]
Reward: -0.3432811662189949
Done: False
Observation: [ 1.23327904e+00 -4.24542640e-01 -4.62028505e-01  2.51245046e-03
 -2.70607241e-01 -3.89414534e-01 -1.64700218e-01 -1.39244616e-01
 -1.34362824e+00 -1.17501753e+00 -1.00000000e+01 -1.00000000e+01
  3.59446832e-01  6.46108742e+00 -1.00000000e+01  1.00000000e+01
 -1.00000000e+01]
Reward: -0.33931137513533964
Done: False
Observation: [ 1.22089413e+00 -5.70626219e-01 -6.29615067e-01  4.28609932e-03
 -1.89206963e-01 -5.81745260e-01 -1.06174637e-01 -2.00538914e-01
 -1.44205114e+00 -1.91791924e+00 -1.00000000e+01 -1.00000000e+01
  1.11102812e-01  1.00000000e+01 -1.00000000e+01  4.39798628e+00
 -2.63945976e+00]
Reward: -0.39837599783335403
Done: False
Observation: [  1.2063032   -0.7017736   -0.77064885  -0.01101738  -0.07527445
  -0.74736977  -0.06857081  -0.21161107  -1.07144761  -1.69406299
 -10.         -10.         -3.85271001  10.         -10.
   5.18162638  -0.68860326]
Reward: -0.24975812860916732
Done: False
Observation: [  1.1926369   -0.81229835  -0.89154683  -0.01732549   0.02183162
  -0.89989525  -0.01279394  -0.22431153  -0.60940852  -1.72067236
 -10.         -10.          2.25864506   9.6669     -10.
   8.80882449  -2.67314593]
Reward: 0.15615655055787764
Done: False
```