NAME: RUPESH DHIRWANI

CLASS: D16AD

ROLL NO: 10

```
!pip install gym
!pip install matplotlib
```

```
Requirement already satisfied: gym in /usr/local/lib/python3.10/dist-packages (0.25.2)
Requirement already satisfied: numpy>=1.18.0 in /usr/local/lib/python3.10/dist-packages (from gym) (1.25.2)
Requirement already satisfied: cloudpickle>=1.2.0 in /usr/local/lib/python3.10/dist-packages (from gym) (2.2.1)
Requirement already satisfied: gym-notices>=0.0.4 in /usr/local/lib/python3.10/dist-packages (from gym) (0.0.8)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-packages (3.7.1)
Requirement already satisfied: contourpy>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (4.50.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: numpy>=1.20 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (1.25.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (24.0)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

```
import gym
from gym import spaces
import numpy as np
import random

class AntWorldEnv(gym.Env):
    def __init__(self):
        self.action_space = spaces.Discrete(4)  # Four possible actions: up, down, left, right
        self.observation_space = spaces.Discrete(25)  # 5x5 grid world
        self.reset()

    def reset(self):
        self.grid_size = 5
        self.ant_position = [0, 0]  # Initial position of the ant
        self.goal_position = [4, 4]  # Goal position
        self.done = False
        return self._get_observation()

    def _get_observation(self):
        return self.ant_position[0] * self.grid_size + self.ant_position[1]

    def step(self, action):
        if action == 0:  # Up
            self.ant_position[0] = max(0, self.ant_position[0] - 1)
        elif action == 1:  # Down
            self.ant_position[0] = min(self.grid_size - 1, self.ant_position[0] + 1)
        elif action == 2:  # Left
            self.ant_position[1] = max(0, self.ant_position[1] - 1)
        elif action == 3:  # Right
            self.ant_position[1] = min(self.grid_size - 1, self.ant_position[1] + 1)

        if self.ant_position == self.goal_position:
            self.done = True
            reward = 1
        else:
            reward = 0

        return self._get_observation(), reward, self.done, {}

class QLearningAgent:
    def __init__(self, epsilon=0.1, alpha=0.5, gamma=0.99):
        self.q_table = np.zeros((25, 4))  # Q-table for 5x5 grid world with 4 possible actions
        self.epsilon = epsilon
        self.alpha = alpha
        self.gamma = gamma

    def choose_action(self, state):
        if random.uniform(0, 1) < self.epsilon:
            return random.randint(0, 3)  # Choose random action
        else:
            return np.argmax(self.q_table[state])  # Choose action with highest Q-value

    def update_q_table(self, state, action, reward, next_state):
        next_max = np.max(self.q_table[next_state])
        self.q_table[state, action] += self.alpha * (reward + self.gamma * next_max - self.q_table[state, action])

if __name__ == "__main__":
```

```
if __name__ == "__main__":
    env = AntWorldEnv()
    agent = QLearningAgent()

    episodes = 10000
    total_reward = 0

    for episode in range(episodes):
        state = env.reset()
        done = False
        episode_reward = 0

        while not done:
            action = agent.choose_action(state)
            next_state, reward, done, _ = env.step(action)
            agent.update_q_table(state, action, reward, next_state)
            state = next_state
            episode_reward += reward

        total_reward += episode_reward

        if episode % 500 == 0:
            print(f"Episode {episode}/{episodes}")
            print("Total Reward:", total_reward)

    print("Training complete.")
    print("Q-Table:")
    print(agent.q_table)
    print("Number of Episodes:", episodes)
```

```
Episode 0/10000
Total Reward: 1
Episode 500/10000
Total Reward: 501
Episode 1000/10000
Total Reward: 1001
Episode 1500/10000
Total Reward: 1501
Episode 2000/10000
Total Reward: 2001
Episode 2500/10000
Total Reward: 2501
Episode 3000/10000
Total Reward: 3001
Episode 3500/10000
Total Reward: 3501
Episode 4000/10000
Total Reward: 4001
Episode 4500/10000
Total Reward: 4501
Episode 5000/10000
Total Reward: 5001
Episode 5500/10000
Total Reward: 5501
Episode 6000/10000
Total Reward: 6001
Episode 6500/10000
Total Reward: 6501
Episode 7000/10000
Total Reward: 7001
Episode 7500/10000
Total Reward: 7501
Episode 8000/10000
Total Reward: 8001
Episode 8500/10000
Total Reward: 8501
Episode 9000/10000
Total Reward: 9001
Episode 9500/10000
Total Reward: 9501
Training complete.
Q-Table:
[[0.92274469 0.93206535 0.92274469 0.93206535]
 [0.93206535 0.94148015 0.92274469 0.94148015]
 [0.94148015 0.95099005 0.93206535 0.95099005]
 [0.95099005 0.96059601 0.94148015 0.96059601]
 [0.96059601 0.970299   0.95099005 0.96059601]
 [0.92274469 0.92274469 0.93206535 0.94148015]
 [0.93206535 0.93206535 0.93206535 0.95099005]
 [0.94148015 0.96059601 0.94148015 0.96059601]
 [0.95099005 0.970299   0.95099005 0.970299  ]
 [0.96059601 0.9801     0.96059601 0.970299  ]
 [0.93206535 0.22837931 0.8074016  0.81434217]
 [0.94148015 0.79658819 0.46137235 0.480298  ]
 [0.95099005 0.94403392 0.92975046 0.970299  ]
 [0.96059601 0.9801     0.96059601 0.9801    ]
 [0.970299   0.99       0.970299   0.9801    ]
 [0.46137235 0.         0.         0.23068617]
```