```
import numpy as np
import gym
import random
```

```
class OfficeEnv(gym.Env):

    def __init__(self):

        self.action_space = 6
        self.state_space = 9
        self.observation_space = 72
        reward = 0
        state = random.randint(0,env.state_space-1)

    def step(self, action):


        state = random.randint(0,env.state_space-1)


        done = True

        info = {}

        return state, reward, done, info

    def reset(self):
        state = random.randint(0,env.state_space-1)
        self.x = 1
        self.y = 3
        reward = 0
        return state
```

```
env = OfficeEnv()
```

```
q_table = np.zeros([9, 6])
print(q_table)

policy = np.zeros([9, 6])
policy = [[-1, 1, -1, -1, -3, 10],
     [1, 1, -1, 1, 5, -2],
     [1, -1, -1, -1, 5, -2],
     [-1, 1, -1, -1, 5, -2],
     [1, 1, 1, 1, 5, -2],
     [1, -1, -1, -1, 5, -2],
     [-1, 1, -1, -1, 5, -2],
     [1, 1, 1, -1, 5, -2],
```

```
      [1, -1, -1, -1, 5, -2]]
```

```
[[0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0.]]
```

```
num_of_episodes = 1000
max_number_of_steps_per_episode = 10

alpha = 0.1
gamma = 0.99

rate_of_exploration = 1
```

```
rewards_all_episodes = []
import random
# CanBot's Q-Learning algorithm
for i in range(num_of_episodes):
    state = env.reset()

    done = False
    rewards = 0
    penalties = 0

    for step in range(max_number_of_steps_per_episode):


        if random.uniform(0,1) > rate_of_exploration:
            action = np.argmax(policy[:, state])
        else:
            action = random.randint(0,env.action_space-1)

        new_state, reward, done, info = env.step(action)

        q_table[state, action] = (1 - alpha) * q_table[state, action] + \
            alpha * (reward + gamma * np.max(q_table[new_state,:]))

        state = new_state
        rewards += reward

        if done == True:
            break



print("Q-table")
```

```
print("Q-table")
print(q_table)
```

```
Q-table
[[2.05515986 2.44360478 2.66510993 2.30427681 2.18000627 2.51933013]
 [2.64692281 2.79867627 2.34253285 2.36878878 2.89615406 2.81770594]
 [2.67721776 2.49446779 3.06817393 1.89780233 2.38296433 2.40432273]
 [2.42920328 2.60711194 2.19562057 2.49826873 2.38783382 2.30975281]
 [2.24268182 2.78709361 2.90534857 2.17441487 2.44233074 2.68913083]
 [2.40899806 2.13898997 2.60943955 2.00291216 2.50598759 2.42412005]
 [2.85272632 2.46190344 1.94230444 2.06306691 2.64046939 2.54060382]
 [2.48663919 2.63604128 2.55583178 2.56327049 2.22851109 2.82457949]
 [2.21959911 2.30355705 1.33493018 2.77354218 1.77638575 2.53391244]]
```