Maze

資管三 A 黃彥程 109403526

Colab 連結:

https://colab.research.google.com/drive/1cKGbKfO5iy12-ELZV04tUcAFAtfWGxfI

Q-table:

Q-table:

(<built-in function all>,	left	right	up	down
0	-9.950000	-8.920394	-18.649427	-8.624899
1	-17.303990	-23.595994	-38.493489	-27.882234
2	-26.849115	-40.625186	-42.088411	-46.206228
3	-42.146161	-46.601393	-46.941646	-42.845239
4	0.000000	0.000000	0.000000	0.000000
5	0.000000	0.000000	0.000000	0.000000
6	-59.317478	-59.314371	-59.631157	-10.617146
7	0.000000	0.000000	0.000000	0.000000
8	-5.000000	-5.000000	-5.000000	-3.131136
9	0.000000	0.000000	0.000000	0.000000
10	-33.940808	-33.651561	-38.180121	-38.261946
11	-31.785843	-24.265055	-32.369384	-31.459364
12	-27.060423	-11.551400	-56.742973	-24.507937
13	-22.760679	-12.954646	-59.849068	-20.084852
14	-21.287457	-11.664075	-57.647511	-59.404084
15	-22.037127	-17.392559	-57.618942	-11.361171
16	-27.310178	-14.051830	-46.603778	-24.059893
17	-16.873079	-10.567957	-57.154459	-19.664449
18	-20.951048	-10.441993	-55.638869	-55.072171
19	-19.975701	-11.254765	-55.113470	-21.675366
20	-21.332389	-55.697109	-54.821282	-11.245849
21	-9.950000	-9.950000	-8.680101	-7.911682
22	0.000000	0.000000	0.000000	0.000000
23	0.000000	0.000000	0.000000	0.000000
24	-45.978131	-47.284999	-44.280183	-47.405051
25	0.000000	0.000000	0.000000	0.000000
26	-5.000000	-2.962000	-5.000000	-5.000000
27	-2.889100	-2.800000	-3.689509	-2.899067
28	-2.965570	-2.800000	-5.000000	-9.590000
29	-3.043000	-5.000000	-2.992765	-5.000000
30	0.000000	0.000000	0.000000	0.000000
31	0.000000	0.000000	0.000000	0.000000
32	-42.628831	-20.085654	-31.637639	-46.093196
33	-16.667223	-13.235470	-11.576683	-16.161839
34	-21.706578	-52.368611	-13.361977	-53.219450
35	0.000000	0.000000	0.000000	0.000000
36	-52.584971	-11.537629	-21.586192	-57.076223
37	-21.430387	-11.527026	-18.198990	-56.391164
38	-21.026993	-57.210796	-11.134133	-55.593109
39	0.000000	0.000000	0.000000	0.000000
40	-32.528664	-25.398433	-19.380495	-27.952799

最佳:

score:5 step:277

```
['Episode 883: total_steps=261 score:2']
['Episode 884: total_steps=217 score:2']
['Episode 885: total_steps=517 score:5']
['Episode 886: total_steps=277 score:5']
['Episode 887: total_steps=419 score:4']
['Episode 888: total_steps=311 score:4']
```

過程:

```
def get_env_feedback(S, A, path):  # S為目前狀態，A為採取之動作，R為反饋reward
    global SCORE
    if A == "right":
        if (S % N_STATES_x == N_STATES_x - 1) or (S + 1  in position):  #走到最右邊不能再走了
            S_ = S
            R = -50
        elif S + 1 in treasure_position:
            S_ = S + 1
            SCORE += 1
            R = 150
        elif S + 1 in path:
            S_ = S + 1
            R = -10
        else:                           #沒有特別的
            S_ = S + 1
            R = -1

    elif A == "left":                   # todo
        if (S % N_STATES_x == 0) or (S - 1 in position):    #走到最左邊不能再走了
            S_ = S
            R = -50
        elif S - 1 in treasure_position:
            S_ = S - 1
            SCORE += 1
            R = 150
        elif S - 1 in path:
            S_ = S - 1
            R = -10
        else:                           #沒有特別的
            S_ = S - 1
            R = -1
```

```
elif A == "up":
    if (S < N_STATES_x) or(S - 21  in position) : #表示在最上面了
        S_ = S
        R = -50
    elif S - 21 in treasure_position:
        S_ = S - 21
        SCORE += 1
        R = 150
    elif S - 21 in path:
        S_ = S - 21
        R = -10
    else:                                #沒有特別的
        S_ = S - 21
        R = -1

elif A == "down":
    if (S // N_STATES_x == N_STATES_y - 1) or (S + 21 in position): #表示在最下面了
        S_ = S
        R = -50
    elif S == GOAL - N_STATES_x:         #終點
        S_ = "terminal"
        R = -1000
    elif S + 21 in treasure_position:
        S_ = S + 21
        SCORE += 1
        R = 150
    elif S + 21 in path:
        S_ = S + 21
        R = -10
    else:                                    #沒有特別的
        S_ = S + 21
        R = -1

return S_, R
```

四個方向方法其實都差不多，就是用他那個方向向前一格後判斷是否是在牆壁上 or 出界(這兩個我寫在一起) or 拿到寶藏又或者是重複走之前走過的路，而成功出去我只有寫在”down”因為只有這種方式出的去，以減少判斷次數。

撞到牆壁或出界就不讓他走，然後扣它分數，這樣下次它就不太會再撞一次，重複走之前的路會浪費時間所以也扣它分，拿到寶藏就 SCORE += 1，然後 reward 給高一點引誘它來拿，具體判斷它有沒有重複拿到我寫在 rl():

```python
def rl():
    global epsilon
    global SCORE
    global treasure_position
    q_table = build_q_table(N_STATES_x, N_STATES_y, ACTIONS)
    for episode in range(MAX_EPISODES):
        treasure_found = set()
        step_counter = 0
        S = 0
        SCORE = 0
        is_terminated = False
        path = []
        treasure_position = [6, 79, 170, 212, 227]
        #epsilon = update_epsilon(epsilon)
        update_env(S, episode, step_counter)
        while not is_terminated:
            A = choose_action(S, q_table)
            path.append(S)
            S_, R = get_env_feedback(S, A, path)
            q_predict = q_table.loc[S, A]
            if S_ in treasure_position:
                treasure_position.remove(S_)
                for i in path[-15: ]:
                  q_table.loc[i, :] = 0
            else:
                if S_ != "terminal":
                    q_target = R + GAMMA * q_table.iloc[S_, :].max()
                else:
                    q_target = R
                    is_terminated = True
                q_table.loc[S, A] += ALPHA * (q_target - q_predict)
            S = S_
            update_env(S, episode, step_counter + 1)
            step_counter += 1
    return all, q_table
```

（treasure_found 我忘記刪掉了，那是之前還在測試的寫法）

拿到一次就從裡面刪掉，然後一個 episode 結束就再重新賦予 treasure_position 裡面的位置。

心得:

這份作業首先要先把地圖放進去，但是我後來是直接在 reward 用判斷的方式，不是真的放位置在地圖中，因為我一開始查資料多

數是有看到人家有真的放位置在圖裡面，後來看了程式碼還是覺得這樣多此一舉，我直接判斷有沒有撞牆比較快，然後就是我一開始在設置有沒有走出迷宮時，設錯方向(因為一開始預設是向右有出去的方法，但是作業地圖是要向下)，然後我就一直不知道為甚麼沒有走出來，十分懊惱。

其實在做這份作業時，一開始當然是只有寫到 reward 部分利用 Reward 比例去鼓勵機器人去拿寶藏跟出去，但是後來發現這樣雖然跑得出來但是步數下降速度不快，然後去研究 Epsilon Greedy 加上了遞減(把作業程式碼的 > EPSILON 改成 <，然後加上遞減)讓機器越來越按照之前學習的方式走，這樣就比較會有學習成效，可是我發現雖然這樣有辦法把步數壓到不超過 100，SCORE 卻是一直歸零狀態，然後我就又設置了如果 SCORE 沒有 5 不給它出去的規定，想當然而直接變成幾萬步在跳。後來看到 google 上有人設置走回頭路就扣分，才又把 path 判斷補上，確實是快了不少，但是一樣 SCORE 都沒有甚麼辦法拿到 5 分，試了好幾次不同組合之後發現關鍵問題是，如果每次想拿寶藏走回頭路又都知道要扣大分，是我我也不拿，才又加上如果拿到寶藏最後 15 筆 Q-table 上的資訊重置，然後把我前面雞婆弄的遞減 EPSILON 改回助教寫的模樣，果然功夫不到家還是不要亂搞比較好，害我弄了好久。