인공지능 Project Report

Report 제목: Project #2

Dynamic Programming

제출기한:

2022년 11월 25일 (금)

~

2022년 12월 09일 (금)

학 과: 컴퓨터정보공학부

담당교수: 박철수 교수님

수강시간: 금요일 3,4교시

학 번: 2018202065

성 명: 박 철 준

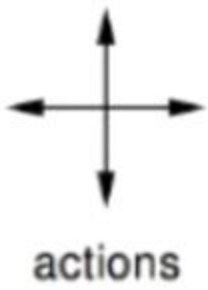
● Introduction

■ 제목

Dynamic Programming

■ 목적

Dynamic Programming(DP)을 이용하여 environment의 model을 푸는 과제이다. DP는 큰 Programming을 여러 process로 나누어 해결한다는 의미이다. 큰 순차적인 행동 결정을 작은 process로 나누어 제시된 grid world에서 Policy iteration과 Value iteration을 policy를 최적화 시켜보는 것이 과제이다. 영역은 7x7의 시작점과 끝점을 가진 gird-world이다. Action은 상하좌우 4가지입니다. 제일 바깥쪽 state에서 gird-world 밖으로 나가는 action을 취할 경우, 제자리로 돌아온다. 중간중간에 함정(reward=-100)이 존재한다.
다음은 grid world의 예시이다.



START	-1	-100	-1	-1	-1	-1
-1	-1	-100	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-100	-100	-1
-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1
-1	-1	-100	-100	-1	-1	END

■ 과제 요구 사항

1. Policy evaluation을 구현한다.
2. 이후 k의 값을 0,1,2,3과 같이 초반의 변하는 부분과 충분히 k의 값을 늘려 수렴하게 된 부분을 모두 캡처하여 보고서에 작성한다
3. Policy Improvement 구현한다.
4. Policy가 업데이트되는 과정과 각각 state에서의 action을 매트릭스로 만들어 보고서에 작성하여야 한다.
5. Value Iteration 구현 k의 값을 0,1,2,3과 같이 초반의 변하는 부분과 충분히 k의 값을 늘려 수렴하게 된 부분을 모두 캡처하여 보고서에 작성하여야 한다.
6. Policy Improvement 구현하여 random policy와 greedy policy와 비교하여 optimal policy를 찾아야한다.

● Algorithm

■ Policy evaluation 구현 **(policy_evaluation.py 파일)**

▶ 설계 방식

아래는 현재 state의 value를 구할 수 있는 즉 Policy evaluation을 할 수 있는 Bellman Expectation equation이다.

$$v_{k+1}(s) = \sum_{a \in A} \pi(a|s) \left(\mathcal{R}_s^a + \gamma \sum_{s' \in S} \mathcal{P}_{ss'}^a v_k(s') \right)$$

이를 통해 구현하였고 iteration과정을 통해 k의 값을 증가시켜 value의 수렴값을 찾았다.

■ Policy Improvement 구현 **(policy_improvement_normally.py 파일 및 policy_improvement_policy_iteration.py 파일)**

해당 방식은 2가지 경우로 나뉘게 되는데 첫 번째로는 optimal value를 Policy evaluation로 구하고 해당 optimal value에 대한 policy로 재정의해주는 방법

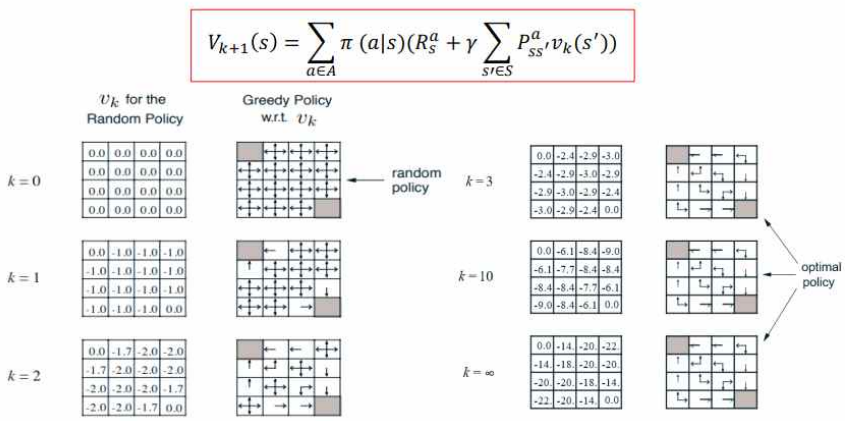
두 번째는 Policy evaluation 과정에서 iteration 할 때마다 나오는 value 값에 대해서 그 값에 해당하는 policy를 반복적으로 즉 policy iteration을 하여 구하는 방법이다.

해당 보고서에서 첫 번째 방식은 policy_imporvement_normally이라고 명하고자 하고 두 번째 방식은 policy_improvement_policy_Iteration로 명하고자 한다.

▶ policy_imporvement_normally 설계 방식 **(policy_improvement_normally.py 파일)**

$$V_{k+1}(s) = \sum_{a \in A} \pi(a|s)(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_k(s'))$$

해당 수식은 Policy evaluation을 할 수 있는 Bellman Expectation equation이다. 우리가 하려는 policy_imporvement는 아래와 같이 4 by 4 grid에서 다음과 같은 과정으로 나타낼 수 있다.



즉 Policy evaluation을 통해 나온 optimal value에 대해서 해당 value의 policy를 찾아주면 optimal policy를 구할 수 있다는 것이다. 하지만 해당 방식은 잘못되었다. 왜냐하면 Policy evaluation의 iteration 과정에서 iteration 한번 할 때마다 value가 생성될 것인데 이때 해당 값의 policy를 구하여 다음 iteration에서 value값을 구할 때 변경한 policy를 사용하여 value를 구하고 이를 반복적으로 수행하여 policy iteration을 해야하기 때문이다. 해당 방식은 policy_improvement_policy_iteration 설계 방식에서 설명하겠다.

▶ policy_improvement_policy_iteration설계 방식 (**policy_improvement_policy_iteration.py 파일)**

policy_improvement_policy_iteration는 아래의 식을 보면 알 수 있듯이

$$V_{k+1}(s) = \sum_{a \in A} \pi^*(a|s)(R_s^a + \gamma \sum_{s' \in S} P_{ss'}^a v_k(s'))$$

기존의 Policy evaluation의 Bellman Expectation equation와는 확연한 차이를 가지고 있는 데 policy를 나타내는 함수가 파이 스타 함수로 바뀌었다는 것이다. 이때 파이 스타는 새로운 policy 즉 improved된 policy에 대한 함수이다. 이것을 구하기 위해서 사용한 방법은 q-function 즉 action – value function이다. 이는 아래와 같다.

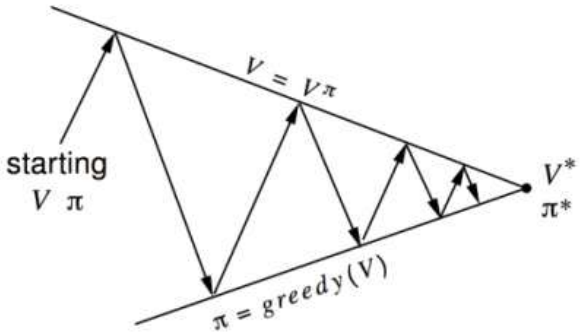
$$q_*(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s')$$

해당 q값을 통하여 더 좋은 action을 선택하는 policy로 만드는 과정을 만들어야 하며 q-func.의 max값만을 취하는 것이 바로 greedy policy improvement이다. 즉 이것의 의미는 현재 state에서 갈 수 있는 state 중 가장 높은 곳으로 이동하는 action을 취하는 것이다. 또한 q값의 max값의 개수는 한 개의 state에서 할 수 있는 action 4개가 있으므로 최대 4개 까지 중복으로 나올 수 있다. 그리하여 중복의 개수를 파악하고 1/중복개수를 통해 policy 확률을 구하게된다. 또한 이렇게 구해진 policy를 사용하여 다음 value를 구하는데 바뀐 policy를 기존에 구현했던 Policy evaluation 알고리즘에 적용하여 value를 구하게 된다. 마지막으로 해당 과정을 iteration하여 policy의 수렴값을 구할 수 있고 이에 따라 value값도 수렴되게 되는 방식이다.

해당 프로그램의 흐름은 다음과 같다.

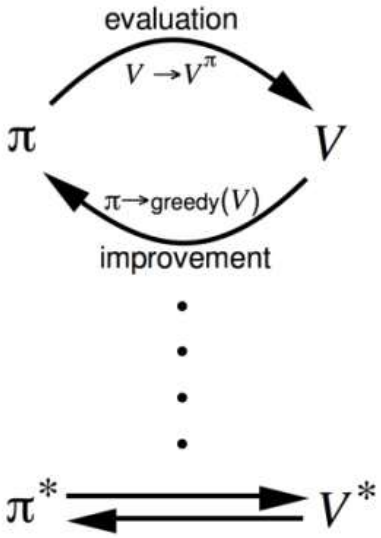
프로그램 실행 초기 policy는 random하게 정의 -> iteration되면 q-func을 통해 q값의 max값을 구하고 이를 바탕으로 파이 함수 재정의하고 이에 따라 policy 변화함 -> 바뀐 policy를 사용하여 Policy evaluation진행 -> 해당과정을 입력받은 iteration 횟수만큼 진행 -> policy와 value의 수렴값을 도출 즉 아래 그림과 같은 모습으로 진행한다.

POLICY ITERATION



Policy evaluation Estimate v_π
Iterative policy evaluation

Policy improvement Generate $\pi' \geq \pi$
Greedy policy improvement



■ Value Iteration 구현 **(value_iteration.py 파일)**

▶ 설계 방식

아래는 value iteration에 대한 Bellman Expectation equation이다.

$$v_*(s) \leftarrow \max_{a \in \mathcal{A}} \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_*(s') \right)$$

식을 통해 알 수 있듯이 앞서 두 방식과는 다른 방식으로 value를 구하게 되는데 이 방식은 policy를 사용하지 않는다는 것이다. 즉 현재 state 기준으로 4가지 방향의 value을 이용하여 구해진 값의 max값을 취하는 것이고 앞서 구현했던 evaluation 과정에서 policy가 관여한 부분을 삭제하고 나오는 값 중 max 값을 취하여 현재 state의 max값을 구하게 된다. 하지만 여기에서 수렴함을 보이기 위해선 모두가 같은 value를 가지는 상황을 만들면 안 된다. 왜냐하면, 수렴을 할 수 없고 발산하는 구조이기 때문이다. 그래서 도착점의 value값을 0으로 고정하여 수렴시킬 수 있도록 해주어야 수렴을 시키기 위해선 iteration을 해야한다. 고로 첫 번째로 구현한 policy evaluation 함수에서 policy를 사용한 부분을 제거하고 위의 식의 가로 부분의 max값을 구해주는 부분을 추가하여 구현하였다.

● Result

■ Policy evaluation 결과 **(policy_evaluation.py 파일)**

▶ 초반 0~3에서 변화하는 부분

```
Iteration: 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.]]

Iteration using random_policy(0.25): 1
[[ -1.   -25.75 -50.5  -25.75  -1.    -1.    -1.  ]
 [ -1.   -25.75 -25.75 -25.75  -1.    -1.    -1.  ]
 [ -1.    -1.   -25.75  -1.   -25.75 -25.75  -1.  ]
 [ -1.    -1.    -1.   -25.75 -25.75 -25.75 -25.75]
 [ -1.    -1.    -1.    -1.   -25.75 -25.75  -1.  ]
 [ -1.    -1.   -25.75 -25.75  -1.    -1.    -1.  ]
 [ -1.   -25.75 -50.5  -50.5  -25.75  -1.     0.  ]]

Iteration using random_policy(0.25): 2
[[ -8.188 -51.5   -82.438 -51.5    -8.188  -2.     -2.   ]
 [ -8.188 -39.125 -57.688 -39.125 -14.375  -8.188  -2.   ]
 [ -2.    -14.375 -32.938 -26.75  -39.125 -39.125 -14.375]
 [ -2.     -2.    -14.375 -32.938 -51.5   -51.5   -39.125]
 [ -2.     -2.     -8.188 -20.562 -39.125 -39.125 -14.375]
 [ -2.    -14.375 -45.312 -45.312 -20.562  -8.188  -1.75 ]
 [ -8.188 -45.312 -88.625 -88.625 -45.312  -7.938   0.   ]]

Iteration using random_policy(0.25): 3
[[ -20.016  -71.063 -111.282  -71.063  -20.016   -6.094   -3.   ]
 [ -15.375  -58.688  -74.156  -63.328  -24.656  -15.375   -7.641]
 [  -7.641  -20.016  -54.047  -37.032  -58.688  -54.047  -24.656]
 [  -3.      -9.188  -20.016  -54.047  -66.422  -67.969  -55.594]
 [  -3.      -7.641  -21.562  -32.391  -58.687  -54.047  -24.594]
 [  -7.641  -24.656  -64.875  -69.515  -35.484  -18.344   -7.078]
 [ -16.922  -64.875 -117.468 -117.468  -66.359  -16.359    0.   ]]
```

value값이 계속해서 변화하고 있다. 수렴하는 양상은 현재 보기 힘듦

▶ k의 값을 늘려 수렴하게 된 부분

3000번의 iteration을 진행하였다.

```
Iteration using random_policy(0.25): 2995
[[-3337.538 -3381.159 -3383.404 -3268.265 -3114.174 -3006.443 -2947.546]
 [-3289.918 -3319.536 -3298.788 -3204.218 -3063.816 -2953.61  -2884.65 ]
 [-3208.68  -3205.281 -3184.996 -3083.005 -2979.262 -2855.531 -2748.796]
 [-3126.842 -3103.912 -3049.911 -2959.546 -2811.698 -2637.457 -2502.209]
 [-3063.936 -3029.614 -2947.19  -2790.571 -2567.529 -2277.39  -2017.374]
 [-3031.352 -2999.42  -2914.665 -2684.018 -2287.459 -1784.201 -1268.525]
 [-3026.7   -3018.05  -2925.031 -2640.379 -2110.089 -1299.43      0.   ]]

Iteration using random_policy(0.25): 2996
[[-3337.538 -3381.159 -3383.404 -3268.265 -3114.174 -3006.443 -2947.546]
 [-3289.918 -3319.536 -3298.788 -3204.218 -3063.816 -2953.61  -2884.65 ]
 [-3208.68  -3205.281 -3184.996 -3083.005 -2979.262 -2855.531 -2748.796]
 [-3126.842 -3103.912 -3049.911 -2959.546 -2811.698 -2637.457 -2502.209]
 [-3063.936 -3029.614 -2947.19  -2790.571 -2567.529 -2277.39  -2017.374]
 [-3031.352 -2999.42  -2914.665 -2684.018 -2287.459 -1784.201 -1268.525]
 [-3026.7   -3018.05  -2925.031 -2640.379 -2110.089 -1299.43      0.   ]]

Iteration using random_policy(0.25): 2997
[[-3337.538 -3381.159 -3383.404 -3268.265 -3114.174 -3006.443 -2947.546]
 [-3289.918 -3319.536 -3298.788 -3204.218 -3063.816 -2953.61  -2884.65 ]
 [-3208.68  -3205.281 -3184.996 -3083.005 -2979.262 -2855.531 -2748.796]
 [-3126.842 -3103.912 -3049.911 -2959.546 -2811.698 -2637.457 -2502.209]
 [-3063.936 -3029.614 -2947.19  -2790.571 -2567.529 -2277.39  -2017.374]
 [-3031.352 -2999.42  -2914.665 -2684.018 -2287.459 -1784.201 -1268.525]
 [-3026.7   -3018.05  -2925.031 -2640.379 -2110.089 -1299.43      0.   ]]

Iteration using random_policy(0.25): 2998
[[-3337.538 -3381.159 -3383.404 -3268.265 -3114.174 -3006.443 -2947.546]
 [-3289.918 -3319.536 -3298.788 -3204.218 -3063.816 -2953.61  -2884.65 ]
 [-3208.68  -3205.281 -3184.996 -3083.005 -2979.262 -2855.531 -2748.796]
 [-3126.842 -3103.912 -3049.911 -2959.546 -2811.698 -2637.457 -2502.209]
 [-3063.936 -3029.614 -2947.19  -2790.571 -2567.529 -2277.39  -2017.374]
 [-3031.352 -2999.42  -2914.665 -2684.018 -2287.459 -1784.201 -1268.525]
 [-3026.7   -3018.05  -2925.031 -2640.379 -2110.089 -1299.43      0.   ]]

Iteration using random_policy(0.25): 2999
[[-3337.538 -3381.159 -3383.404 -3268.265 -3114.174 -3006.443 -2947.546]
 [-3289.918 -3319.536 -3298.788 -3204.218 -3063.816 -2953.61  -2884.65 ]
 [-3208.68  -3205.281 -3184.996 -3083.005 -2979.262 -2855.531 -2748.796]
 [-3126.842 -3103.912 -3049.911 -2959.546 -2811.698 -2637.457 -2502.209]
 [-3063.936 -3029.614 -2947.19  -2790.571 -2567.529 -2277.39  -2017.374]
 [-3031.352 -2999.42  -2914.665 -2684.018 -2287.459 -1784.201 -1268.525]
 [-3026.7   -3018.05  -2925.031 -2640.379 -2110.089 -1299.43      0.   ]]

Iteration using random_policy(0.25): 3000
[[-3337.538 -3381.159 -3383.404 -3268.265 -3114.174 -3006.443 -2947.546]
 [-3289.918 -3319.536 -3298.788 -3204.218 -3063.816 -2953.61  -2884.65 ]
 [-3208.68  -3205.281 -3184.996 -3083.005 -2979.262 -2855.531 -2748.796]
 [-3126.842 -3103.912 -3049.911 -2959.546 -2811.698 -2637.457 -2502.209]
 [-3063.936 -3029.614 -2947.19  -2790.571 -2567.529 -2277.39  -2017.374]
 [-3031.352 -2999.42  -2914.665 -2684.018 -2287.459 -1784.201 -1268.525]
 [-3026.7   -3018.05  -2925.031 -2640.379 -2110.089 -1299.43      0.   ]]
```

value값을 보게 되면 iteration 횟수가 증가해도 변화가 없다 즉 수렴함을 알 수 있다.

■ Policy Improvement 결과 **(policy_improvement_normally.py 파일 및 policy_improvement_policy_iteration.py 파일)**

▶ policy_imporvement_normally 결과 **(policy_improvement_normally.py 파일)**

- 초반 0~3에서 변화하는 부분

```
Iteration: 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.]]

Iteration using random_policy(0.25): 1
[[ -1.   -25.75 -50.5  -25.75  -1.    -1.    -1.  ]
 [ -1.   -25.75 -25.75 -25.75  -1.    -1.    -1.  ]
 [ -1.    -1.   -25.75  -1.   -25.75 -25.75  -1.  ]
 [ -1.    -1.    -1.   -25.75 -25.75 -25.75 -25.75]
 [ -1.    -1.    -1.    -1.   -25.75 -25.75  -1.  ]
 [ -1.    -1.   -25.75 -25.75  -1.    -1.    -1.  ]
 [ -1.   -25.75 -50.5  -50.5  -25.75  -1.     0.  ]]

at each state, chosen action is :
[['←↑↓ ' '←   ' '←↓→ ' '→   ' '↑↓→ ' '←↑↓→' '←↑↓→']
 ['←↑↓ ' '↓←  ' '←↓→ ' '↓→  ' '↑→  ' '←↑→ ' '←↑↓→']
 ['←↑↓→' '↓←  ' '←↓→ ' '←↑↓→' '↑←  ' '↑→  ' '↑→  ']
 ['←↑↓→' '←↑↓→' '↓←  ' '←↑↓ ' '←↑↓→' '←↑↓→' '↑↓  ']
 ['←↑↓→' '←↑↓→' '←↑→ ' '←   ' '↓←  ' '↓→  ' '↓→  ']
 ['←↑↓→' '↑←  ' '↑←  ' '↑→  ' '→   ' '←↓→ ' '↓   ']
 ['←↑↓ ' '↑←  ' '↑←  ' '↑→  ' '↑→  ' '→   ' 'T    ']]
```

```
Iteration using random_policy(0.25): 2
[[ -8.188 -51.5   -82.438 -51.5    -8.188  -2.     -2.   ]
 [ -8.188 -39.125 -57.688 -39.125 -14.375  -8.188  -2.   ]
 [ -2.    -14.375 -32.938 -26.75  -39.125 -39.125 -14.375]
 [ -2.     -2.    -14.375 -32.938 -51.5   -51.5   -39.125]
 [ -2.     -2.     -8.188 -20.562 -39.125 -39.125 -14.375]
 [ -2.    -14.375 -45.312 -45.312 -20.562  -8.188  -1.75 ]
 [ -8.188 -45.312 -88.625 -88.625 -45.312  -7.938   0.   ]]

at each state, chosen action is :
[['←↑↓ ' '←   ' '←→  ' '→   ' '→   ' '↑→  ' '←↑↓→']
 ['↓   ' '←   ' '↓   ' '→   ' '↑→  ' '↑→  ' '↑→  ']
 ['↓←  ' '↓←  ' '↓←  ' '↓←  ' '↑   ' '↑   ' '↑   ']
 ['←↑↓→' '↓←  ' '←   ' '←   ' '←   ' '↑↓→ ' '↑↓  ']
 ['←↑↓→' '↑←  ' '←   ' '←   ' '↓←  ' '↓   ' '↓   ']
 ['↑←  ' '↑←  ' '↑   ' '↑→  ' '→   ' '→   ' '↓   ']
 ['↑   ' '←   ' '↑←  ' '↑→  ' '→   ' '→   ' 'T    ']]
```

```
Iteration using random_policy(0.25): 3
[[ -20.016  -71.063 -111.282  -71.063  -20.016   -6.094   -3.   ]
 [ -15.375  -58.688  -74.156  -63.328  -24.656  -15.375   -7.641]
 [  -7.641  -20.016  -54.047  -37.032  -58.688  -54.047  -24.656]
 [  -3.      -9.188  -20.016  -54.047  -66.422  -67.969  -55.594]
 [  -3.      -7.641  -21.562  -32.391  -58.687  -54.047  -24.594]
 [  -7.641  -24.656  -64.875  -69.515  -35.484  -18.344   -7.078]
 [ -16.922  -64.875 -117.468 -117.468  -66.359  -16.359    0.   ]]

at each state, chosen action is :
[['↓   ' '←   ' '↔  ' '→   ' '→   ' '→   ' '↑→  ']
 ['↓   ' '←   ' '↓   ' '→   ' '→   ' '↑   ' '↑   ']
 ['↓   ' '←   ' '↓←  ' '↓←  ' '↑   ' '↑   ' '↑   ']
 ['↓←  ' '←   ' '←   ' '←   ' '←   ' '↑↓  ' '↓   ']
 ['↑←  ' '←   ' '←   ' '←   ' '←   ' '↓   ' '↓   ']
 ['↑   ' '↑←  ' '↑   ' '↑   ' '→   ' '→   ' '↓   ']
 ['↑   ' '←   ' '↑←  ' '→   ' '→   ' '→   ' 'T    ']]
```

optimal policy가 계속해서 변하므로 수렴 양상은 보이지 않으므로 iteration을 더 반복시켜야 함을 알 수 있다.

- k의 값을 충분히 늘려 변화하는 부분

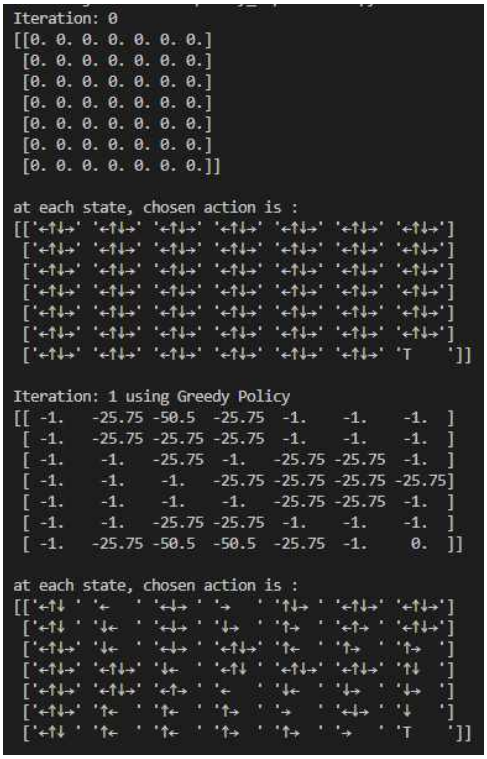
3000번의 진행을 통해 optimal_value를 구하고 해당 부분의 policy를 구하면 아래와 같은 결과를 보인다.

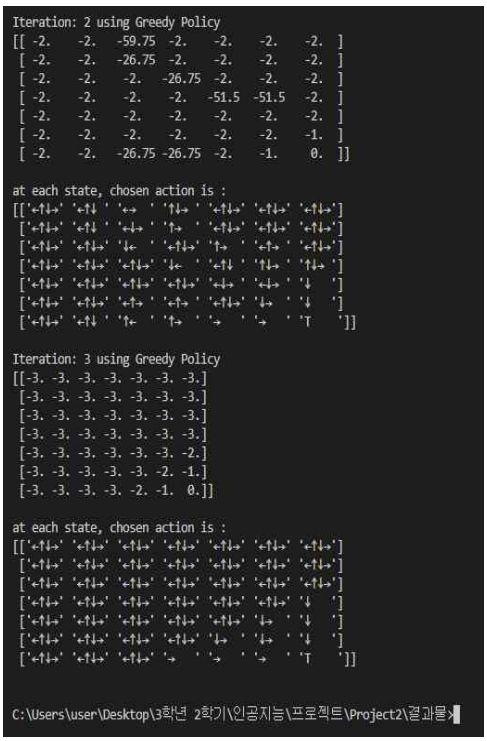
```
Iteration using random_policy(0.25): 3000
[[-3337.538 -3381.159 -3383.404 -3268.265 -3114.174 -3006.443 -2947.546]
 [-3289.918 -3319.536 -3298.788 -3204.218 -3063.816 -2953.61  -2884.65 ]
 [-3208.68  -3205.281 -3184.996 -3083.005 -2979.262 -2855.531 -2748.796]
 [-3126.842 -3103.912 -3049.911 -2959.546 -2811.698 -2637.457 -2502.209]
 [-3063.936 -3029.614 -2947.19  -2790.571 -2567.529 -2277.39  -2017.374]
 [-3031.352 -2999.42  -2914.665 -2684.018 -2287.459 -1784.201 -1268.525]
 [-3026.7   -3018.05  -2925.031 -2640.379 -2110.089 -1299.43      0.   ]]

at each state, chosen action is :
[['↓   ' '↓   ' '→   ' '→   ' '→   ' '→   ' '↓   ']
 ['↓   ' '↓   ' '↓   ' '→   ' '→   ' '↓   ' '↓   ']
 ['↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ']
 ['↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '→   ' '↓   ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '→   ' '→   ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '→   ' '→   ' 'T    ']]
```

optimal_policy가 잘 구해짐을 알 수 있다. 하지만 해당 방식의 결과는 잘못되었다. 왜냐하면, Policy evaluation의 iteration 과정에서 iteration 한번 할 때마다 value가 생성될 것인데 이때 해당 값의 policy를 구하여 다음 iteration에서 value값을 구할 때 변경한 policy를 사용하여 value를 구하고 이를 반복적으로 수행하여 policy iteration을 해야하기 때문이다. 해당 방식은 policy_improvement_policy_iteration 결과에서 설명하겠다.

▶ policy_improvement_policy_iteration 결과 (**policy_improvement_policy_iteration.py 파일)**

- 초반 0~3에서 변화하는 부분

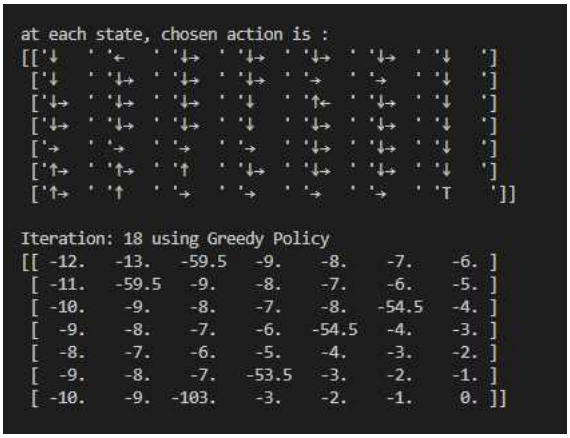


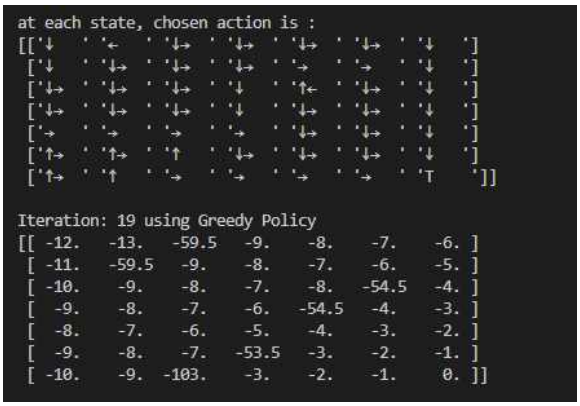


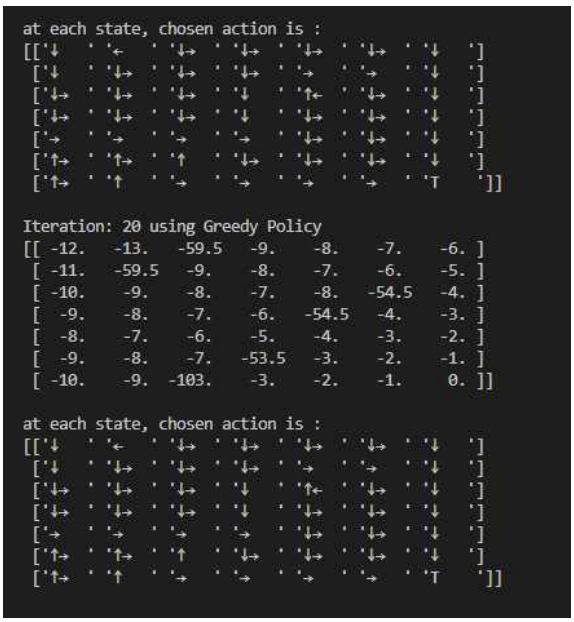
아직 까진 policy가 수렴하는 양상을 보기 힘들며 이를 통해 value값도 수렴하는 양상은 보기 힘듦 하지만 변경된 policy에 의해 value값이 잘 반영됨을 알 수 있다.

- k의 값을 충분히 늘려 변화하는 부분

20번의 iteration을 진행하였다.







policy값을 보게 되면 iteration 횟수가 증가해도 변화가 없다 즉 수렴함을 알 수 있다. value값 또한 이렇게 수렴한 policy로 인해 변화가 없다는 것을 알 수 있고 이는 수렴함을 알 수 있다. 또한 state에서 action을 즉 policy를 매트릭스로 만들어 화살표로 출력하였다.

■ Value Iteration 결과 **(value_iteration.py 파일)**

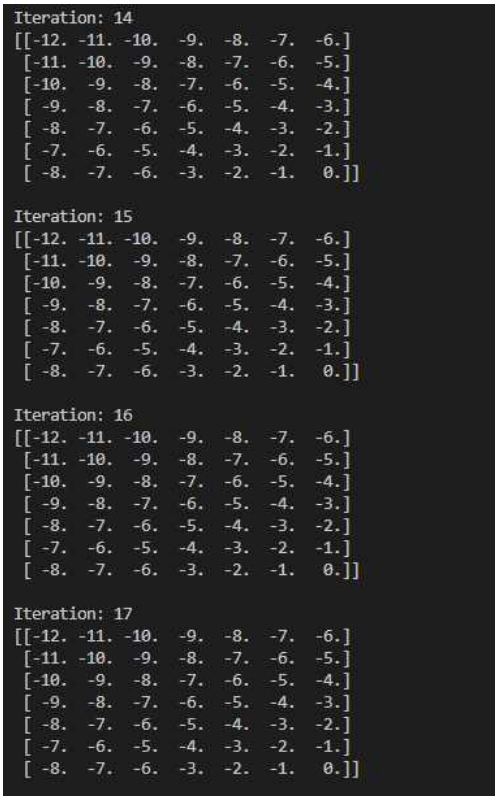
▶ 초반 0~3에서 변화하는 부분

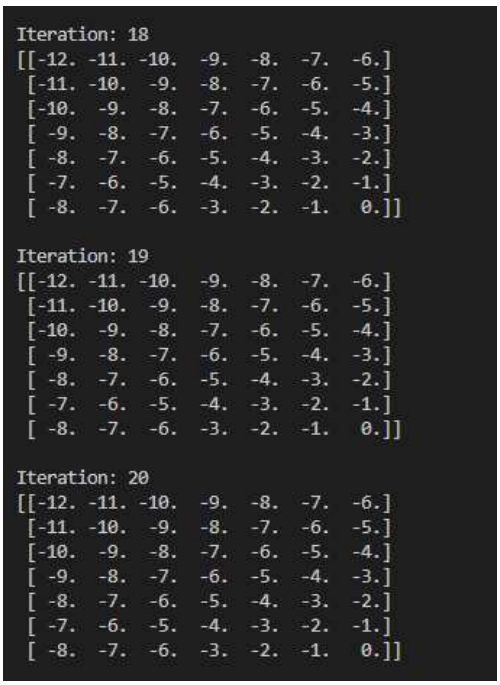
```
Iteration: 0
[[0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]]

Iteration: 1
[[-1. -1. -1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1. -1. -1.]
 [-1. -1. -1. -1. -1. -1.  0.]]

Iteration: 2
[[-2. -2. -2. -2. -2. -2. -2.]
 [-2. -2. -2. -2. -2. -2. -2.]
 [-2. -2. -2. -2. -2. -2. -2.]
 [-2. -2. -2. -2. -2. -2. -2.]
 [-2. -2. -2. -2. -2. -2. -2.]
 [-2. -2. -2. -2. -2. -2. -1.]
 [-2. -2. -2. -2. -2. -1.  0.]]

Iteration: 3
[[-3. -3. -3. -3. -3. -3. -3.]
 [-3. -3. -3. -3. -3. -3. -3.]
 [-3. -3. -3. -3. -3. -3. -3.]
 [-3. -3. -3. -3. -3. -3. -3.]
 [-3. -3. -3. -3. -3. -3. -2.]
 [-3. -3. -3. -3. -3. -2. -1.]
 [-3. -3. -3. -3. -2. -1.  0.]]
```

value 값이 변하며 아직까진 수렴하는 모습은 알 수 없다.

▶ k의 값을 충분히 늘려 변화하는 부분

20번의 iteration을 진행하였다.





iteration 횟수가 증가해도 value값을 일정하며 수렴함을 알 수 있다.

■ Policy Improvement 구현하여 random policy와 greedy policy와 비교하여 optimal policy 찾기

  ▶ random policy

```
Iteration: 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.]]

at each state, chosen action is :
[['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' 'T    ']]

Iteration using random_policy(0.25): 1
[[ -1.   -25.75 -50.5  -25.75  -1.    -1.    -1.  ]
 [ -1.   -25.75 -25.75 -25.75  -1.    -1.    -1.  ]
 [ -1.    -1.   -25.75  -1.   -25.75 -25.75  -1.  ]
 [ -1.    -1.    -1.   -25.75 -25.75 -25.75 -25.75]
 [ -1.    -1.    -1.    -1.   -25.75 -25.75  -1.  ]
 [ -1.    -1.   -25.75 -25.75  -1.    -1.    -1.  ]
 [ -1.   -25.75 -50.5  -50.5  -25.75  -1.     0.  ]]

at each state, chosen action is :
[['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' 'T    ']]

Iteration using random_policy(0.25): 2
[[ -8.188 -51.5   -82.438 -51.5    -8.188  -2.     -2.   ]
 [ -8.188 -39.125 -57.688 -39.125 -14.375  -8.188  -2.   ]
 [ -2.    -14.375 -32.938 -26.75  -39.125 -39.125 -14.375]
 [ -2.     -2.    -14.375 -32.938 -51.5   -51.5   -39.125]
 [ -2.     -2.     -8.188 -20.562 -39.125 -39.125 -14.375]
 [ -2.    -14.375 -45.312 -45.312 -20.562  -8.188  -1.75 ]
 [ -8.188 -45.312 -88.625 -88.625 -45.312  -7.938   0.   ]]

at each state, chosen action is :
[['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' '←↑↓→' 'T    ']]
```

4가지 방향으로 갈 policy가 다 같으므로 각 0.25씩 즉 화살표로 표현하면 4방향을 모두 가리키고 있다. 이는 iteration을 진행해도 같다. 이유는 policy upadate를 하지 않았기 때문이다.

▶ greedy policy (**policy_improvement_normally.py 파일**)을 사용하였습니다.

```
Iteration: 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.]]

Iteration using random_policy(0.25): 1
[[ -1.   -25.75 -50.5  -25.75  -1.    -1.    -1.  ]
 [ -1.   -25.75 -25.75 -25.75  -1.    -1.    -1.  ]
 [ -1.    -1.   -25.75  -1.   -25.75 -25.75  -1.  ]
 [ -1.    -1.    -1.   -25.75 -25.75 -25.75 -25.75]
 [ -1.    -1.    -1.    -1.   -25.75 -25.75  -1.  ]
 [ -1.    -1.   -25.75 -25.75  -1.    -1.    -1.  ]
 [ -1.   -25.75 -50.5  -50.5  -25.75  -1.     0.  ]]

at each state, chosen action is :
[['←↑↓' '←  ' '←↓→' '→  ' '↑↓→' '←↑↓→' '←↑↓→']
 ['←↑↓' '↓←  ' '←↓→' '↓→  ' '↑→  ' '←↑→' '←↑↓→']
 ['←↑↓→' '↓←  ' '←↓→' '←↑↓→' '↑←  ' '↑→  ' '↑→  ']
 ['←↑↓→' '←↑↓→' '↓←  ' '←↑↓' '←↑↓→' '←↑↓→' '↑↓  ']
 ['←↑↓→' '←↑↓→' '←↑→' '←  ' '↓←  ' '↓→  ' '↓→  ']
 ['←↑↓→' '↑←  ' '↑←  ' '↑→  ' '→  ' '←↓→' '↓  ']
 ['←↑↓' '↑←  ' '↑←  ' '↑→  ' '↑→  ' '→  ' 'T  ']]
```

```
Iteration using random_policy(0.25): 2
[[ -8.188 -51.5   -82.438 -51.5   -8.188  -2.    -2.   ]
 [ -8.188 -39.125 -57.688 -39.125 -14.375  -8.188  -2.   ]
 [ -2.    -14.375 -32.938 -26.75  -39.125 -39.125 -14.375]
 [ -2.     -2.    -14.375 -32.938 -51.5   -51.5   -39.125]
 [ -2.     -2.     -8.188 -20.562 -39.125 -39.125 -14.375]
 [ -2.    -14.375 -45.312 -45.312 -20.562  -8.188  -1.75 ]
 [ -8.188 -45.312 -88.625 -88.625 -45.312  -7.938   0.   ]]

at each state, chosen action is :
[['←↑↓' '←  ' '←↓→' '→  ' '→  ' '↑→  ' '←↑↓→']
 ['↓  ' '←  ' '↓  ' '→  ' '↑→  ' '↑→  ' '↑→  ']
 ['↓←  ' '↓←  ' '↓←  ' '↓←  ' '↑  ' '↑  ' '↑  ']
 ['←↑↓→' '↓←  ' '←  ' '←  ' '←  ' '↑↓→' '↑↓  ']
 ['←↑↓→' '↑←  ' '←  ' '←  ' '↓←  ' '↓  ' '↓  ']
 ['↑←  ' '↑←  ' '↑  ' '↑→  ' '→  ' '→  ' '↓  ']
 ['↑  ' '←  ' '↑←  ' '↑→  ' '→  ' '→  ' 'T  ']]
```

```
Iteration using random_policy(0.25): 3
[[ -20.016  -71.063 -111.282  -71.063  -20.016   -6.094   -3.   ]
 [ -15.375  -58.688  -74.156  -63.328  -24.656  -15.375   -7.641]
 [  -7.641  -20.016  -54.047  -37.032  -58.688  -54.047  -24.656]
 [  -3.      -9.188  -20.016  -54.047  -66.422  -67.969  -55.594]
 [  -3.      -7.641  -21.562  -32.391  -58.687  -54.047  -24.594]
 [  -7.641  -24.656  -64.875  -69.515  -35.484  -18.344   -7.078]
 [ -16.922  -64.875 -117.468 -117.468  -66.359  -16.359    0.   ]]

at each state, chosen action is :
[['↓   ' '←   ' '↔  ' '→   ' '→   ' '→   ' '↑→  ']
 ['↓   ' '←   ' '↓   ' '→   ' '→   ' '↑   ' '↑   ']
 ['↓   ' '←   ' '↓←  ' '↓←  ' '↑   ' '↑   ' '↑   ']
 ['↓←  ' '←   ' '←   ' '←   ' '←   ' '↑↓  ' '↓   ']
 ['↑←  ' '←   ' '←   ' '←   ' '←   ' '↓   ' '↓   ']
 ['↑   ' '↑←  ' '↑   ' '↑   ' '→   ' '→   ' '↓   ']
 ['↑   ' '←   ' '↑←  ' '→   ' '→   ' '→   ' 'T    ']]
```

```
Iteration using random_policy(0.25): 3000
[[-3337.538 -3381.159 -3383.404 -3268.265 -3114.174 -3006.443 -2947.546]
 [-3289.918 -3319.536 -3298.788 -3204.218 -3063.816 -2953.61  -2884.65 ]
 [-3208.68  -3205.281 -3184.996 -3083.005 -2979.262 -2855.531 -2748.796]
 [-3126.842 -3103.912 -3049.911 -2959.546 -2811.698 -2637.457 -2502.209]
 [-3063.936 -3029.614 -2947.19  -2790.571 -2567.529 -2277.39  -2017.374]
 [-3031.352 -2999.42  -2914.665 -2684.018 -2287.459 -1784.201 -1268.525]
 [-3026.7   -3018.05  -2925.031 -2640.379 -2110.089 -1299.43      0.   ]]

at each state, chosen action is :
[['↓   ' '↓   ' '→   ' '→   ' '→   ' '→   ' '↓   ']
 ['↓   ' '↓   ' '↓   ' '→   ' '→   ' '↓   ' '↓   ']
 ['↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ']
 ['↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '→   ' '↓   ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '→   ' '→   ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '→   ' '→   ' 'T    ']]
```

iteration을 진행함에 따라 policy와 value가 변화하다가 충분히 많은 3000번의 수행으로 수렴함을 알 수 있고 이를 화살표로 나타낸 모습은 아래의 사진과 같다.

```
at each state, chosen action is :
[['↓   ' '↓   ' '→   ' '→   ' '→   ' '→   ' '↓   ']
 ['↓   ' '↓   ' '↓   ' '→   ' '→   ' '↓   ' '↓   ']
 ['↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ']
 ['↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '→   ' '↓   ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '→   ' '→   ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '→   ' '→   ' 'T    ']]
```

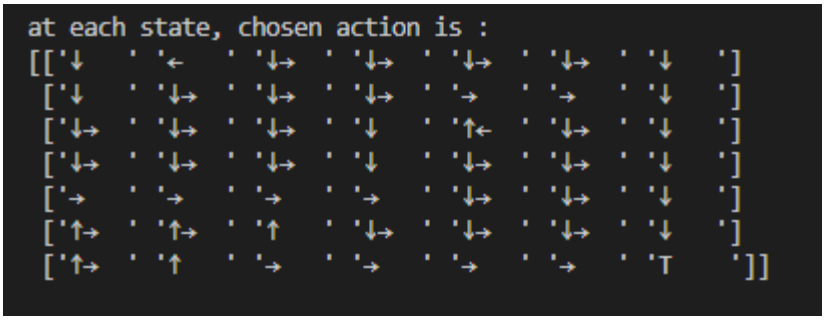
▶ greedy policy **(policy_improvement_policy_iteration.py 파일)**을 사용하였습니다.

```
at each state, chosen action is :
[['←↑↓→' '↑←  ' '↓→  ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
 ['↓    ' '↓→  ' '↓→  ' '↓→  ' '→   ' '→   ' '↓   ']
 ['↓→   ' '↓→  ' '↓→  ' '↓   ' '↑←  ' '↓→  ' '↓   ']
 ['↓→   ' '↓→  ' '↓→  ' '↓   ' '↓→  ' '↓→  ' '↓   ']
 ['→    ' '→   ' '→   ' '→   ' '↓→  ' '↓→  ' '↓   ']
 ['↑→   ' '↑→  ' '↑   ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
 ['↑→   ' '↑   ' '→   ' '→   ' '→   ' '→   ' 'T    ']]

Iteration: 12 using Greedy Policy
[[ -12.   -12.   -59.5   -9.    -8.    -7.    -6. ]
 [ -11.   -59.5   -9.    -8.    -7.    -6.    -5. ]
 [ -10.    -9.    -8.    -7.    -8.   -54.5   -4. ]
 [  -9.    -8.    -7.    -6.   -54.5   -4.    -3. ]
 [  -8.    -7.    -6.    -5.    -4.    -3.    -2. ]
 [  -9.    -8.    -7.   -53.5   -3.    -2.    -1. ]
 [ -10.    -9.  -103.    -3.    -2.    -1.     0. ]]

at each state, chosen action is :
[['↓    ' '↑←  ' '↓→  ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
 ['↓    ' '↓→  ' '↓→  ' '↓→  ' '→   ' '→   ' '↓   ']
 ['↓→   ' '↓→  ' '↓→  ' '↓   ' '↑←  ' '↓→  ' '↓   ']
 ['↓→   ' '↓→  ' '↓→  ' '↓   ' '↓→  ' '↓→  ' '↓   ']
 ['→    ' '→   ' '→   ' '→   ' '↓→  ' '↓→  ' '↓   ']
 ['↑→   ' '↑→  ' '↑   ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
 ['↑→   ' '↑   ' '→   ' '→   ' '→   ' '→   ' 'T    ']]

Iteration: 13 using Greedy Policy
[[ -12.   -13.   -59.5   -9.    -8.    -7.    -6. ]
 [ -11.   -59.5   -9.    -8.    -7.    -6.    -5. ]
 [ -10.    -9.    -8.    -7.    -8.   -54.5   -4. ]
 [  -9.    -8.    -7.    -6.   -54.5   -4.    -3. ]
 [  -8.    -7.    -6.    -5.    -4.    -3.    -2. ]
 [  -9.    -8.    -7.   -53.5   -3.    -2.    -1. ]
 [ -10.    -9.  -103.    -3.    -2.    -1.     0. ]]

at each state, chosen action is :
[['↓    ' '←   ' '↓→  ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
 ['↓    ' '↓→  ' '↓→  ' '↓→  ' '→   ' '→   ' '↓   ']
 ['↓→   ' '↓→  ' '↓→  ' '↓   ' '↑←  ' '↓→  ' '↓   ']
 ['↓→   ' '↓→  ' '↓→  ' '↓   ' '↓→  ' '↓→  ' '↓   ']
 ['→    ' '→   ' '→   ' '→   ' '↓→  ' '↓→  ' '↓   ']
 ['↑→   ' '↑→  ' '↑   ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
 ['↑→   ' '↑   ' '→   ' '→   ' '→   ' '→   ' 'T    ']]
```

```
Iteration: 14 using Greedy Policy
[[ -12.   -13.   -59.5   -9.    -8.    -7.    -6. ]
 [ -11.   -59.5   -9.    -8.    -7.    -6.    -5. ]
 [ -10.    -9.    -8.    -7.    -8.   -54.5   -4. ]
 [  -9.    -8.    -7.    -6.   -54.5   -4.    -3. ]
 [  -8.    -7.    -6.    -5.    -4.    -3.    -2. ]
 [  -9.    -8.    -7.   -53.5   -3.    -2.    -1. ]
 [ -10.    -9.  -103.    -3.    -2.    -1.     0. ]]

at each state, chosen action is :
[['↓   ' '←   ' '↓→  ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
 ['↓   ' '↓→  ' '↓→  ' '↓→  ' '→   ' '→   ' '↓   ']
 ['↓→  ' '↓→  ' '↓→  ' '↓   ' '↑←  ' '↓→  ' '↓   ']
 ['↓→  ' '↓→  ' '↓→  ' '↓   ' '↓→  ' '↓→  ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '↓→  ' '↓→  ' '↓   ']
 ['↑→  ' '↑→  ' '↑   ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
 ['↑→  ' '↑   ' '→   ' '→   ' '→   ' '→   ' 'T    ']]

Iteration: 15 using Greedy Policy
[[ -12.   -13.   -59.5   -9.    -8.    -7.    -6. ]
 [ -11.   -59.5   -9.    -8.    -7.    -6.    -5. ]
 [ -10.    -9.    -8.    -7.    -8.   -54.5   -4. ]
 [  -9.    -8.    -7.    -6.   -54.5   -4.    -3. ]
 [  -8.    -7.    -6.    -5.    -4.    -3.    -2. ]
 [  -9.    -8.    -7.   -53.5   -3.    -2.    -1. ]
 [ -10.    -9.  -103.    -3.    -2.    -1.     0. ]]

at each state, chosen action is :
[['↓   ' '←   ' '↓→  ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
 ['↓   ' '↓→  ' '↓→  ' '↓→  ' '→   ' '→   ' '↓   ']
 ['↓→  ' '↓→  ' '↓→  ' '↓   ' '↑←  ' '↓→  ' '↓   ']
 ['↓→  ' '↓→  ' '↓→  ' '↓   ' '↓→  ' '↓→  ' '↓   ']
 ['→   ' '→   ' '→   ' '→   ' '↓→  ' '↓→  ' '↓   ']
 ['↑→  ' '↑→  ' '↑   ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
 ['↑→  ' '↑   ' '→   ' '→   ' '→   ' '→   ' 'T    ']]

Iteration: 16 using Greedy Policy
[[ -12.   -13.   -59.5   -9.    -8.    -7.    -6. ]
 [ -11.   -59.5   -9.    -8.    -7.    -6.    -5. ]
 [ -10.    -9.    -8.    -7.    -8.   -54.5   -4. ]
 [  -9.    -8.    -7.    -6.   -54.5   -4.    -3. ]
 [  -8.    -7.    -6.    -5.    -4.    -3.    -2. ]
 [  -9.    -8.    -7.   -53.5   -3.    -2.    -1. ]
 [ -10.    -9.  -103.    -3.    -2.    -1.     0. ]]
```

```
 at each state, chosen action is :
 [['↓   ' '←   ' '↓→  ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
  ['↓   ' '↓→  ' '↓→  ' '↓→  ' '→   ' '→   ' '↓   ']
  ['↓→  ' '↓→  ' '↓→  ' '↓   ' '↑←  ' '↓→  ' '↓   ']
  ['↓→  ' '↓→  ' '↓→  ' '↓   ' '↓→  ' '↓→  ' '↓   ']
  ['→   ' '→   ' '→   ' '→   ' '↓→  ' '↓→  ' '↓   ']
  ['↑→  ' '↑→  ' '↑   ' '↓→  ' '↓→  ' '↓→  ' '↓   ']
  ['↑→  ' '↑   ' '→   ' '→   ' '→   ' '→   ' 'T    ']]
```

iteration을 진행함에 따라 policy와 value가 변화하다가 13번 이상부터는 두 값의 변화가 없기 때문에 optimal policy를 구했음을 알 수 있고 이를 화살표로 나타낸 모습은 아래와 같다.



● Consideration

이번 과제를 진행하면서 가장 어려웠던 부분은 policy Improvement를 구현함에 policy_iteration할 경우 변화하는 policy를 어떻게 구할 것인지에 대한 부분이었으며 이를 해결하고자 수업에서 배웠던 action-value function 즉 q-function을 사용하여 max q값의 경우에서 파이 값이 1이 되고 이것이 한 state에서 취할 수 있는 action의 계수로 나눈 것이 파이 스타라는 것을 알게 되었고 이러한 방식을 코드로 구현하였고 policy를 재정의할 수 있었다 이후 만들었던 policy_evaluation함수에 변경된 policy로 value값을 구하는 것으로 policy_improvement_policy_iteration 방식을 구현하였다.