인공지능시스템 HW 1

제출기한 : 2023년 4월 11일 23:59

작성한 프로그램 코드 및 실행 결과 캡처본을 포함하여 word로 작성
작성된 Programming 코드 (.py 또는 ipynb) 파일 함께 제출
word파일과 코드 파일을 하나의 .zip파일로 압축하여 제출
제출 파일명 형식 : 홍길동_211234_HW01.zip (미준수 시 감점)
본인이 작성한 모든 프로그래밍 코드에는 주석을 상세히 달아 자세히 설명 (누락 시 감점)
COPY 적발 시 해당 과제는 0점 처리
제출 딜레이 : 1주 당 25%씩 점수 감점

1. 신입생에게 전대마을 소개하기 (10점)

올해 입학한 23학번 신입생은 전대마을이 어떻게 생겼는지 궁금하여 방문하게 되었습니다. 하지만 아무도 없이 혼자 온 신입생은 혼자 두리번거리다가 마침 전대마을 앞을 지나가던 당신에게 도움을 받기로 결정합니다. 또한 마침 당신은 인공지능시스템 수업에서 **깊이 우선 탐색 (Depth First Search) 알고리즘**을 배웠습니다.
이를 활용하여 빈 코드를 채우고 23학번 신입생에게 전대마을에 존재하는 집과 창고가 몇 개인지 알려주세요.
- 전대마을은 N X M (N, M은 4 이상 10 이하의 정수) 크기의 직사각형으로 이루어져 있습니다.
- 0은 도로, 1은 집을 의미합니다.
- 하나의 집은 둘 이상의 1로 구성되어 있으며 분리되어 있지 않고 모두 연결되어 있습니다.
- 즉, 주변에 위치한 8개 중 하나라도 1이 존재하면 이는 같은 집을 의미합니다.
- 집과 집 사이는 모두 분리되어 있습니다.
- 한 개의 1로 구성된 곳은 집이 아닌 창고입니다.

Ex) 아래 마을에서는 총 4개의 집, 1개의 창고가 존재합니다.

1	1	0	0	1	0
1	0	0	1	0	1
0	0	0	0	0	0
0	1	0	1	0	1 창고
1	1	1	0	0	0
1	0	0	0	1	1

```
#######
코드작성
#######

# 전대마을(행렬) 입력
n_row, n_col = map(int, input('마을의 행과 열 개수를 공백 기준으로 분리하여 입력: ').split())
village = []
for i in range(n_row):
    col = list(map(int, input(f'{i} 행 입력: ').split()))
    village.append(col)

# 마을에 있는 집 개수 출력
#### 코드 작성 ####
print(f'전대마을에는 {#작성}개의 집, {#작성}개의 창고가 있습니다.')
```

입출력 예시 #1

```
마을의 행과 열을 공백을 기준으로 분리하여 입력: 6 6
0 행 입력: 1 1 0 0 1 0
1 행 입력: 1 0 0 1 0 1
2 행 입력: 0 0 0 0 0 0
3 행 입력: 0 1 0 1 0 1
4 행 입력: 1 1 1 0 0 0
5 행 입력: 1 0 0 0 1 1
전대마을에는 4개의 집, 1개의 창고가 있습니다.
```

입출력 예시 #2

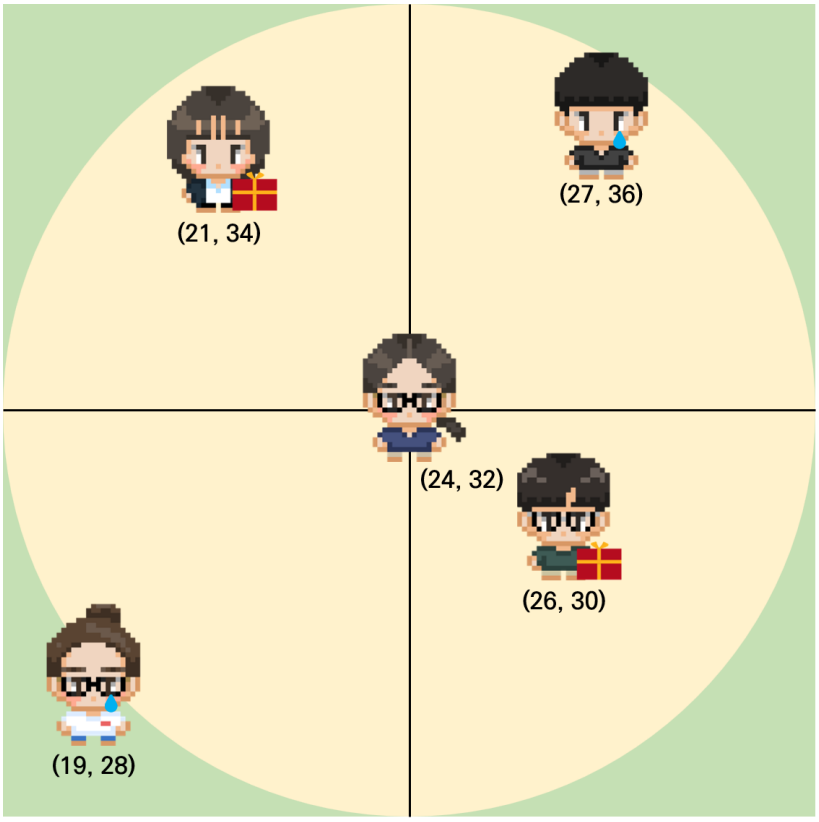
```
마을의 행과 열을 공백을 기준으로 분리하여 입력: 4 5
0 행 입력: 1 1 0 0 1
1 행 입력: 1 0 0 0 1
2 행 입력: 0 0 0 1 0
3 행 입력: 1 1 0 0 0
전대마을에는 3개의 집, 0개의 창고가 있습니다.
```

2. 인공지능학부 디저트 나눔 사건 (15 점)

2023 년 X 월 X 일 인공지능학부에서 학생들에게 맛있는 디저트를 나눠주기로 하였습니다. 하지만 디저트를 나눠준다는 소식이 퍼져 인공지능학부 학생 외 타 학과 학생들도 몰래 받으러 와버렸습니다. 인공지능학부 예산으로 나눠주는 만큼 디저트를 나눠주는 임무를 맡은 당신은 인공지능학부 학생들에게만 디저트를 나눠주기로 결정합니다. 다행히 마침 인공지능시스템 수업에서 **너비 우선 탐색 (Breadth First Search) 알고리즘**을 배웠습니다. BFS 알고리즘을 활용하여 빈 코드를 채우고 인공지능학부 학생들에게 디저트를 나눠주고 총 몇 명의 학생들이 디저트를 전달받았는지 확인해주세요.

- 입력으로 원의 중심 좌표(cx, cy)와 반지름(radius)이 들어갑니다. (cx, cy: 정수 / radius: 2 이상 10 이하의 정수)
- 당신은 현재 원의 중심 좌표(cx, cy)에 서 있으며 모든 정수 좌표에 학생들이 서 있습니다.
- 디저트를 받을 수 있는 영역은 당신을 중심으로 반지름 radius 인 원 내부입니다. (접점 포함 X)
- 디저트 전달 시작점인 당신은 상하좌우 총 네 사람에 대해서만 학생을 확인하고 인공지능학부 학생에게만 디저트를 전달합니다.
- 디저트를 전달받은 인공지능학부 학생들은 동일하게 상하좌우 총 네 사람에 대해서 학생을 탐색하고 같은 인공지능학부 학생에게 디저트를 전달할 수 있습니다.
- 아쉽게도, 상하좌우 네 사람 모두 타 학과 학생들에게 둘러싸인 인공지능학부 학생은 디저트를 받을 수 없습니다.
- 주어진 영역 밖 인공지능학부 학생 또한 디저트를 받을 수 없습니다.
- 인공지능학부 학생 판별은 다음과 같습니다.
  > 확인하려는 학생 좌표값을 각각 절대값으로 취한 후, 각 자릿수를 분리하여 합해줍니다.
  Ex) (25, 34) → 2+5+3+4=14 / (-14, 8) → 1+4+8=13
  > 총합이 16 이하이거나 짝수라면 인공지능학부 학생입니다. 아닌 경우 타 학과 학생입니다.

Ex) 중심 좌표 (24, 32), 반지름 6 일 때 인공지능학부 학생 4 명에 관한 결과입니다.



인공지능학부 학생 A
(24, 32) → (23, 32) → (22, 32)
→ (21, 32) → (21, 33) → (21, 34)
디저트 수령 완료!

인공지능학부 학생 B
주변 네 방향에 대한 학생 모두 타 학과
(26, 36), (28, 36), (27, 37), (27, 35)
디저트 수령 불가능

인공지능학부 학생 C
주어진 영역이 아닌 영역 밖에 위치
디저트 수령 불가능

인공지능학부 학생 D
(24, 32) → (25, 32) → (26, 32)
→ (26, 31) → (26, 30)
디저트 수령 완료!

```
#######
코드작성
#######

# 원의 중심좌표와 가로, 세로 길이 입력
cx, cy = map(int, input('중심 좌표 입력: ').split())
radius = int(input('원의 반지름 입력: '))

#### 코드작성 ####

print(f'디저트를 받은 인공지능학부 학생은 총 {answer}명입니다.')
```

입출력 예시 #1

중심 좌표 입력: 24 32
원의 반지름 입력: 6
디저트를 받은 인공지능학부 학생은 총 85명입니다.

입출력 예시 #2

중심 좌표 입력: 0 0
원의 반지름 입력: 4
디저트를 받은 인공지능학부 학생은 총 44명입니다.

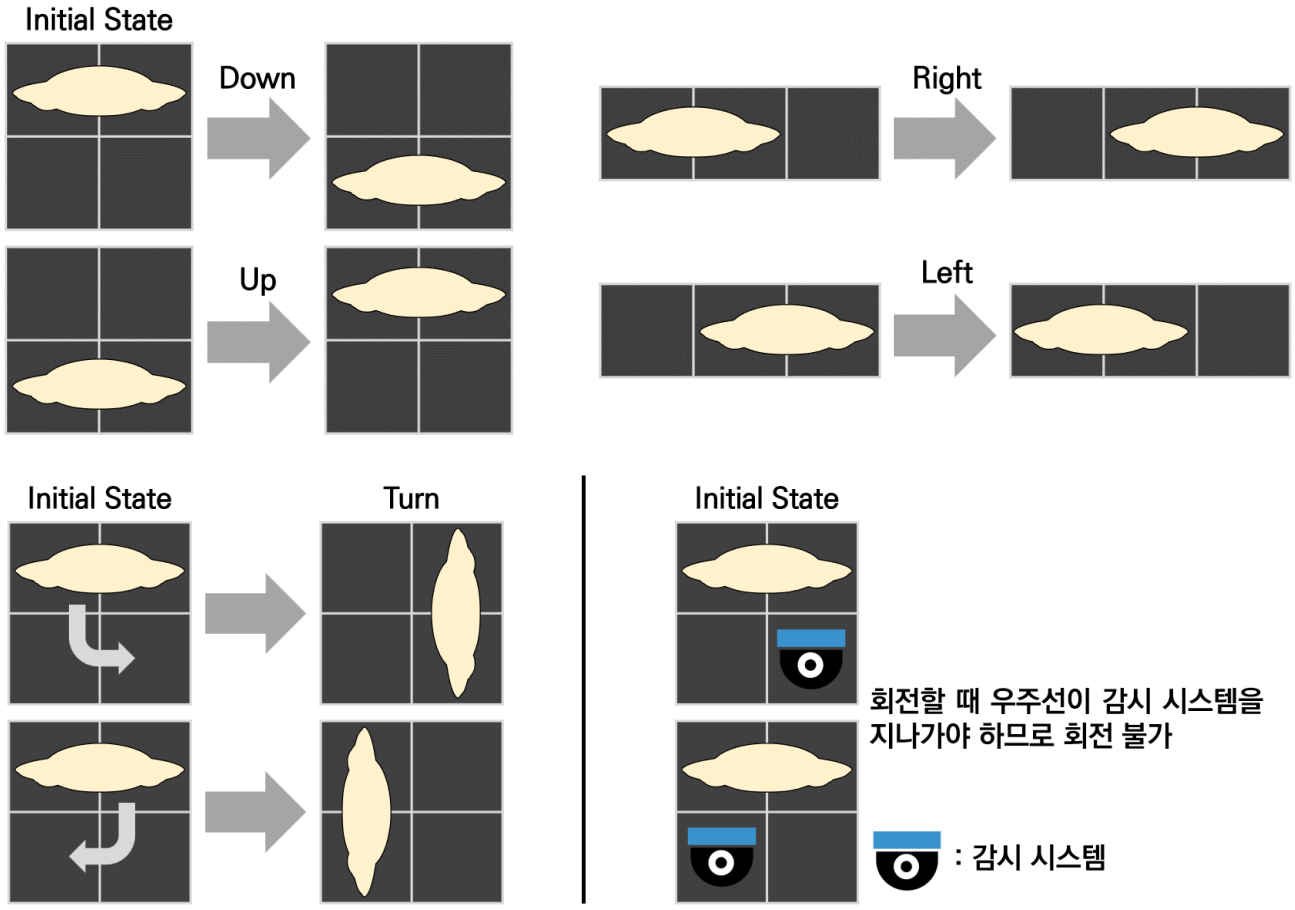
입출력 예시 #3

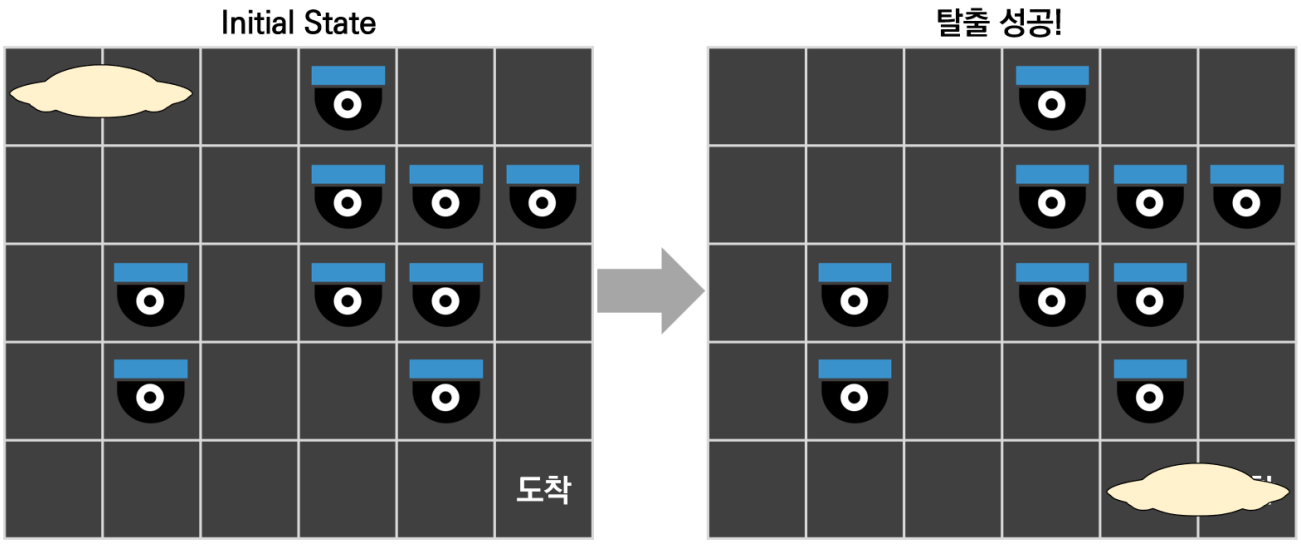
중심 좌표 입력: -17 28
원의 반지름 입력: 5
디저트를 받은 인공지능학부 학생은 총 0명입니다.

3. 감시 시스템을 피해 탈출하기 (20 점)

당신과 친구들은 따뜻한 날씨에 봄을 즐기기 위해 봉지에 모여 봉플을 즐기고 있습니다. 그러던 중, 마실 물과 간식을 사러 가기 위해 당신은 제 1 학생회관 매점으로 가게 되었습니다. 물과 과자를 구매하고 봉지로 돌아가려던 당신은 친구들이 외계인에게 납치당하는 현장을 목격해버리고 맙니다. 외계인들은 당신이 물을 사러 간 사이에 당신의 친구들을 납치하여 우주선에 태우고 도망가버린 것이었습니다. 다행히 목적지에 대한 지도가 봉지에 떨어져 있는 것을 발견한 당신은 곧장 당신만의 우주선을 타고 목적지로 향했습니다. 우주선을 타고 목적지로 향한 당신은 친구들이 갇힌 곳을 발견하게 되고 곧장 문을 따고 친구들과 함께 탈출하기로 하였습니다. 하지만 당신이 친구들과 탈출하려는 것을 감지한 외계인들이 감시 시스템을 발동하였습니다. 감시 시스템에 걸리게 되면 당신도 같이 잡혀버리게 되므로 당신은 알고리즘을 설계하여 감시 시스템을 피해 최대한 안전하고 빠르게 탈출하기로 합니다. 당신이 배운 **탐색 알고리즘을 활용**하여 우주선의 탈출 알고리즘을 설계하여 감시 공간을 탈출할 수 있는 최단 시간을 알려주세요. (풀이 과정 자세히 작성 요망)

- 우주선은 1 X 2 크기로 감시 공간에서 총 두 칸을 차지합니다.
- 감시 공간은 N X M 크기이며, 1 은 감시 시스템을, 0 은 탈출 가능 경로를 의미합니다.
- 출발지는 (0, 0)과 (0, 1)이며 도착지는 (N-1, M-1)입니다. (N, M 은 4 이상, 10 이하의 정수)
- 우주선은 해당 공간 밖으로 나갈 수 없으며 또한 감시 시스템이 있는 곳으로 갈 수 없습니다.
- 우주선은 상하좌우 총 네 방향에 대해 한 칸씩 이동할 수 있습니다. (그림 참조)
- 우주선은 우주선이 차지하는 두 칸 중 어느 칸이든 상관없이 한 칸을 기준으로 모든 방향으로 90 도씩 회전할 수 있습니다. 우주선을 회전할 때에 회전하는 방향에는 감시 시스템이 존재하면 안됩니다. (그림 참조)
- 우주선이 이동하는 경우나 회전하는 경우 모두 한 번 움직일 때마다 1 초 소요됩니다.
- 우주선의 한 부분(한 칸)이라도 도착지에 닿으면 탈출 성공입니다.
- 출발지는 항상 감시 시스템이 존재하지 않습니다.






```
#######
코드작성
#######

# 감시 공간 크기 입력
n, m = map(int, input('감시 공간 크기 입력: ').split())

board = []
for s in range(n): # 감시 공간 입력
    tmp = list(map(int, input(f'{s}행 입력: ').split()))

#### 코드 작성 ####

print(f'감시 공간을 탈출할 수 있는 최단 시간은 {answer}초입니다.')
```

```
입출력 예시 #1

감시 공간 크기 입력: 5 6
0행 입력: 0 0 0 1 0 0
1행 입력: 0 0 0 1 1 1
2행 입력: 0 1 0 1 1 0
3행 입력: 0 1 0 0 1 0
4행 입력: 0 0 0 0 0 0
감시 공간을 탈출할 수 있는 최단 시간은 8초입니다.
```

4. 틱택토: 절대 지지 않는 (항상 비기거나 승리하는) 틱택토 인공지능 만들기 (25 점)

- 코드는 ttt_main.ipynb 또는 ttt_main.py 를 활용하며, computer() 함수 및 minimax() 함수를 재구현 하시오.
- 함수내부 동작을 완전히 새롭게 구현하시오. 함수의 input/output 을 바꿔도 되지만, 함수명은 변경 불가
- 반드시 본인이 새롭게 작성한 코드에는 상세한 주석을 첨부하시오. 누락 시 감점
- copy 엄금. 인터넷 외부 자료를 참고할경우 동일한 자료를 참고한 학생들끼리 copy checker 에 적발됩니다. 외부 자료를 통해 아이디어는 얻을 수 있지만, 실제 구현은 본인이 스스로 해주시길 바랍니다.

실행예) ▨ : **빈곳 / ○ : 사용자 / × 컴퓨터**

```
게임 시작
▨▨▨
▨▨▨
▨▨▨
============= Your turn =============
돌을 놓을 행을 입력하세요 (0~2):0
돌을 놓을 열을 입력하세요 (0~2):0
○▨▨
▨▨▨
▨▨▨
============= Computer =============
○▨▨
▨▨×
▨▨▨
============= Your turn =============
돌을 놓을 행을 입력하세요 (0~2):2
돌을 놓을 열을 입력하세요 (0~2):2
○▨▨
▨▨×
▨▨○
============= Computer =============
○▨▨
▨▨×
×▨○
============= Your turn =============
돌을 놓을 행을 입력하세요 (0~2):1
돌을 놓을 열을 입력하세요 (0~2):0
○▨▨
○▨×
×▨○
============= Computer =============
○▨▨
○××
×▨○
============= Your turn =============
돌을 놓을 행을 입력하세요 (0~2):2
돌을 놓을 열을 입력하세요 (0~2):1
○▨▨
○××
×○○
============= Computer =============
○▨×
○××
×○○
컴퓨터가 승리하였습니다.
```

5. 구름게임 : 컴퓨터는 항상 최선을 다한다. (30 점)

구름게임은 N 차 (N 은 3 이상 5 이하의 정수) 정방행렬을 가진 맵 안에서 사용자와 컴퓨터 간 게임이다. 맵은 기본적으로 낭떠러지(0)과 구름(1)으로 이루어져 있다. 플레이어들은 자신의 위치에서 상하좌우 총 네 방향에 대해서만 움직일 수 있으며 맵 밖으로, 혹은 낭떠러지로 이동은 불가하다. 즉, 구름인 곳에 대해서만 이동 가능하며 시작 위치 또한 구름 위치에서만 가능하다. 구름은 한 번 밟고 지나가면 없어져 낭떠러지로 변한다. 사용자와 컴퓨터는 한 번씩 턴을 주고받으며 이동하게 되며 컴퓨터가 먼저 시작한다. 상대방이 더 이상 이동할 수 없을 경우 이기게 된다.

컴퓨터는 이길 수 있는 수가 있는 경우 그 위치로 움직인다. 만약 지게 되는 수라 하더라도 최선을 다해 최대한 늦게 지는 쪽을 택한다.

MiniMax 알고리즘으로 인공지능을 갖춘 컴퓨터가 구름게임 안에서 최선을 다하도록 설계하세요.

- 제공한 기본 코드 파일(cloud_main.py 또는 cloud_main.ipynb)
  의 #### 코드 작성 #### 부분을 채워 코드를 완성시킬 것
- 작성한 코드에 대해 상세한 주석 필수 (누락 시 감점)
- **MiniMax 알고리즘과 alpha-beta 가지치기 모두 구현할 것**
- 필요하다면 함수 추가 작성 가능

  (풀이 과정 자세히 작성 요망)

Ex) 구름게임 예시