

백준 21610번 마법사 상어와 비바라기

소프트웨어학과 19학번 한지성

격자의 가장 왼쪽 윗 칸은 (1, 1)이고, 가장 오른쪽 아랫 칸은 (N, N)이다. 마법사 상어는 연습을 위해 1번 행과 N번 행을 연결했고, 1번 열과 N번 열도 연결했다. 즉, N번 행의 아래에는 1번 행이, 1번 행의 위에는 N번 행이 있고, 1번 열의 왼쪽에는 N번 열이, N번 열의 오른쪽에는 1번 열이 있다.

=> 0,0 (빨강) 에서 <-로 2칸 가면 0,1 (파랑)

Ex) $N=3$, $3 \times 3 \Rightarrow (N+2) \times (N+2)$, 5×5

1,1 (0,0)	1,2 (0,1)	1,3 (0,2)
2,1 (1,0)	2,2 (1,1)	2,3 (1,2)
3,1 (2,0)	3,2 (2,1)	3,3 (2,2)

// 처음에는 $N=3$ 이면 위 아래 왼쪽 오른쪽 다 붙여서 3×3 을 -> 5×5 라 생각했는데 문제 그림보고 이해함..

각 칸 물 바꾸니 행렬

```
water_basket = [list(map(int, input().split())) for _ in range(N)]
```

⇒NxN 만듦

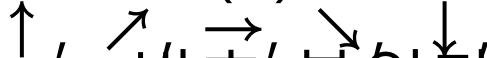
비바라기를 시전하면 (N, 1), (N, 2), (N-1, 1), (N-1, 2)에 비구름이 생긴다. 구름은 칸 전체를 차지한다. -> 비가 내리는 칸들

```
cloud= [(N-1, 0), (N-1, 1), (N-2, 0), (N-2, 1)]
```

이제 구름에 이동을 M번 명령하려고 한다. => 총 반복 횟수

i번째 이동 명령은 방향 d_i 과 거리 s_i 로 이루어져 있다.

=> d랑 s 입력 받음

방향은 총 8개의 방향(d)이 있으며, 8개의 정수로 표현한다. 1부터 순서대로 진행된다.  이다. 이동을 명령하면 다음은 순서대로 번호 붙인다

$1 \leq d_i \leq 8$, -> 인덱스 0에 0,0 채운다!

$\leftarrow(1), \nwarrow(2), \uparrow(3), \nearrow(4), \rightarrow(5), \searrow(6), \downarrow(7), \swarrow(8)$

$\Rightarrow (0,-1), (-1,-1), (-1,0), (-1,1), (0,1), (1,1), (1,0), (1,-1)$

-1, -1	-1, 0	-1, 1
0, -1	0, 0	0, 1
1, -1	1, 0	1, 1

$dx = [0, 0, -1, -1, -1, 0, 1, 1, 1]$

$dy = [0, -1, -1, 0, 1, 1, 1, 0, -1]$

1. 모든 구름이 d_i 방향으로 s_i 칸 이동한다.

d 와 s 를 입력 받은거로 M 번 이동시킨다 => 한 개씩 대입, M 번 반복
비 내리는 영역 저장한 값 하나씩 d, s 이용해 거기에 비 내려 (+1)

구름 자리 인덱스로 보기 -> for i in $(\text{len}(\text{cloud}))$: ~~

$(N-1, 0), (N-1, 1), (N-2, 0), (N-2, 1)$

Ex) $N=3, (N-1, 0), d$ 가 1(\leftarrow)(0, -1), 이고 s 가 2일 때,
 $x = 2, y = 0, dx[d], dy[d]$ 넣고 2칸 가야 된다. -> 더하고 곱하고 다 해봤다
끝과 끝 연결되니까 $\%N$ $a=bq+r (0 \leq r < |b|)$

$x + dx[d] * s$ 하면 $2 + 0 * 2 = 2, 2\%3 = 2$

$y + dy[d] * s$ 하면 $0 + -1 * 2 = -2, -2\%3 = 1 \# 2 = 0*3 + 2, -2 = -3 + 1$

$(2, 0)$ 빨강 -> $(2, 1)$ 파랑

$\Rightarrow (x + dx[d] * s) \% N, (y + dy[d] * s) \% N$

그리고 여기 +1해 그리고 이 위치 뒤에 저장 (반복)
-> 저장해서 물버그에 활용 $\text{cloud2} = []$

1,1 (0,0)	1,2 (0,1)	1,3 (0,2)
2,1 (1,0)	2,2 (1,1)	2,3 (1,2)
3,1 (2,0)	3,2 (2,1)	3,3 (2,2)

$$x2, y2 = (x + dx[d] * s) \% N, (y + dy[d] * s) \% N$$

구글링 해보니까 $a=bq+r$ ($0 \leq r \leq |b|$)로 음수 나눌 때 몫 나머지

C랑 Python이랑 다르다

C와 Python의 음수 나머지 연산 방법의 차이

C는 버림, Python은 내림

3. 구름이 모두 사라진다. => +1 해주고 지우기 cloud = []

4. 2에서 물이 증가한 칸 (r, c)에 물 복사버그 마법을 시전한다.
칸 (r,c) cloud2에 저장한거 사용

물복사버그 마법을 사용하면, 대각선 방향으로 거리가 1인 칸에 물이 있는 바구니의 수만큼 (r, c)에 있는 바구니의 물이 양이 증가한다.

이때는 이동과 다르게 경계를 넘어가는 칸은 대각선 방향으로 거리가 1인 칸이 아니다.

↖(2), ↗(4), ↘(6), ↙(8) 에 행렬 값이 있는지 확인 -> 범위보고 판단?

카운트 변수에 저장하고 개수 만큼 + 해줘

range(2, 9, 2) -> 2, 4, 6, 8 됐네

If 문으로 칸이 있는지(x in range(N) and y in range(N))랑 바구니 물 양이 0이상인지 검사 -> True면 카운트 총 카운트값 칸(r,c)에 더해준다

5. 바구니에 저장된 물의 양이 2 이상인 모든 칸에 구름이 생기고, 물의 양이 2 줄어든다. 이때 구름이 생기는 칸은 3에서 구름이 사라진 칸이 아니어야 한다.

3이 최종 구름 위치니까 저장한 위치는 구름 $x \rightarrow \text{cloud2}$ 로 확인
2 이상인 칸 확인하고 그 자리에 구름 생성(cloud에 넣기)이랑 -2해주기

지금까지 이걸 총 M번 반복..

=====

➔ M번의 이동이 모두 끝난 후 바구니에 들어있는 물의 양의 합을 구해보자.

행렬(NxN)에 있는 값 반복문으로 다 더하기

```
answer = 0
```

```
for x in range(N):
```

```
    for y in range(N):
```

```
        answer += water_basket[x][y]
```

```
print(answer)
```

이제 코드

```
N, M = map(int, input().split())
```

```
dx = [0, 0, -1, -1, -1, 0, 1, 1, 1]
```

```
dy = [0, -1, -1, 0, 1, 1, 1, 0, -1]
```

```
cloud = [(N-1, 0), (N-1, 1), (N-2, 0), (N-2, 1)]
```

```
water_basket = [list(map(int, input().split())) for _ in range(N)]
```

```
moves = [list(map(int, input().split())) for _ in range(M)]
```

```
for d, s in moves:
```

```
    cloud2 = []
```

```
    for i in range(len(cloud)):
```

```
        x, y = cloud[i][0], cloud[i][1]
```

```
        x2, y2 = (x + dx[d] * s) % N, (y + dy[d] * s) % N
```

```
        water_basket[x2][y2] += 1
```

```
        cloud2.append((x2, y2))
```

```
    |
```

```
    cloud = []
```

```
for x, y in cloud2:
```

```
    cnt = 0
```

```
    for d in range(2, 9, 2):
```

```
        x2, y2 = x + dx[d], y + dy[d]
```

```
        if (x2 in range(N) and y2 in range(N)) and water_basket[x2][y2] >= 0 :
```

```
            cnt += 1
```

```
    water_basket[x][y] += cnt
```

```

for x in range(N):
    for y in range(N):
        if water_basket[x][y] >=2 and (x,y) not in cloud2:
            cloud.append((x,y))
            water_basket[x][y] -= 2

```

```

answer = 0
for x in range(N):
    for y in range(N):
        answer += water_basket[x][y]

print(answer)

```

```

5 4
0 0 1 0 2
2 3 2 1 0
4 3 2 9 0
1 0 2 9 0
8 8 2 1 0
1 3
3 4
8 1
4 8

```

97

틀렸어 틀렸어 ~

결국 모든 코드에 print()문 넣고 하나씩 값 확인

```
print()
```

```
print()
```

```
....
```

```
print()
```

```
print()
```

```
.....
```

어디서 틀린걸까

문제내용이랑 코드(print()문) 계속 보던 중...

버그 관련 코드에서 값이 맞지 않고 문제 내용에서 발견한 문장

: 물복사버그 마법을 사용하면, 대각선 방향으로 거리가 1인 칸에

물이 있는 바구니의 수만큼 (r, c)에 있는 바구니의 물이 양이 증가한다.

$0 \leq A[r][c] \leq 100$ 이거 보고 ≥ 0 했는데...

물이 있다 -> 1이상 -> `water_basket[x2][y2] > 0`

if (x2 in range(N) and y2 in range(N)) and `water_basket[x2][y2] >= 0` :

if (x2 in range(N) and y2 in range(N)) and `water_basket[x2][y2] > 0` :

문제를 잘 읽자 ^ ^

```
37         answer
38
39     print(answer)
```

```
5 4
0 0 1 0 2
2 3 2 1 0
4 3 2 9 0
1 0 2 9 0
8 8 2 1 0
1 3
3 4
8 1
4 8
77
```

제출 번호	아이디	문제	결과	메모리	시간	언어	코드 길이	제출한 시간
29438025	hanjise0ng	 21610	맞았습니다!!	126928 KB	448 ms	PyPy3 / 수정	1119 B	1분 전

문제 풀이 시간: 시간 꼭 채움