오후 11:46 9월 4일 금요일

뒤로 미로1_bfs_2.py

```python
sys.stdin = open("input.txt","r")


#반복구조의 bfs
def bfs():
    # 시작점 찾고,
    r = 0
    c = 0
    for i in range(L):
        for j in range(L):
            if maze[i][j] == 2:
                r,c = i,j
                break

    dr = [-1,1,0,0]
    dc = [0,0,-1,1]

    visited = [[0] * L for _ in range(L)]
    queue = list()
    queue.append((r,c,0))
    while queue:
        #시작점으로 부터 위치정보를 얻어올때는,
        #함께 위치 방문에 값이 받아와야함.
        cr,cc,length = queue.pop(0)
        visited[cr][cc] = 1

        for i in range(4):
            nr = cr + dr[i]
            nc = cc + dc[i]
            #범위 내에 있으며, 방문하지 않았으면
            if 0<= nr <L and 0<= nc < L and not visited[nr][nc]:
                if maze[nr][nc] == 0:
                    queue.append((nr,nc,length+1))
                elif maze[nr][nc] == 3:
                    return length+1

    return 0


T = 10
L = 16
for t in range(T):
    tc = input()
    maze = [list(map(int,input())) for _ in range(L)]
    result = bfs()
    print("#{} {}".format(tc,result))
```

N만큼 받고 미로 row

미로 시작점 찾기 (2일 때 시작점)

i,j → r,c로 지정

[2]시작점 찾으면 2이상 탐색멈추고 바로나오기

상하좌우

visited = [[0] * L for _ in range(L)] 방문했는지 안했는지 판단하기 위한 미로 배열과 기본조건은 같지만 초기값은 0으로 해야함

queue = list() 큐 만들기

queue.append((r,c,0)) 큐에 넣기

이동횟수 처음은 0

while queue: 큐가 없을때

cr,cc,length = queue.pop(0) 큐에서 빼기

visited[cr][cc] = 1 방문했으니 1로 표시

for i in range(4): 사방향

nr = cr + dr[i] 사방향에 방문도 있음

nc = cc + dc[i]

if 0<= nr <L and 0<= nc < L and not visited[nr][nc]: 사방으로 다음 방문했는지

if maze[nr][nc] == 0: 방문하지 않았으며 길이면 큐에 넣기

queue.append((nr,nc,length+1)) visited에 표시하고 이동거리 +1 (방문했음)

elif maze[nr][nc] == 3: 도착하면 거리 리턴

return length+1 도착하면 거리 리턴

return 0 길이 없으면 0 리턴

T = 10 → 미로 개수, 테스트 케이스

L = 16 → 미로 가로, 세로 크기

for t in range(T): 미로 배열 모양 만들기 값을 받아오기

result = bfs() 함수실행

https://swexpertacademy.com/main/code/problem/problemDetail.do?contestProbId=AVI4vXUqAGMCFAYD

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	2	0	0	1	0	0	0	0	0	0	0	0	0	1	1
1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1
1	0	0	0	0	0	1	0	1	0	1	0	1	0	1	1
1	1	1	1	1	1	1	0	1	1	1	0	1	0	1	1
1	0	0	0	1	0	0	0	1	0	0	0	0	0	1	1
1	0	1	0	1	0	1	0	1	0	1	1	1	1	1	1
1	0	1	0	0	0	1	0	1	0	0	0	1	0	1	1
1	0	1	1	1	1	1	0	1	1	1	0	1	0	1	1
1	0	0	0	1	0	0	0	1	0	1	0	0	0	1	1
1	1	1	0	1	0	1	1	1	0	1	1	1	1	1	1
1	0	1	0	1	0	0	0	0	0	1	3	0	0	1	1
1	0	1	0	1	1	1	1	1	1	1	1	1	0	1	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	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

Q흐름 (1,1,0) → (1,1,0) → 상하좌우 → (2,1,1), (1,2,1)

① ② ① ②

append pop append

→ (2,1,1), (1,2,1) → (1,2,1) (3,1,2) → (1,2,1) (3,1,2)

pop pop

→ (3,1,2) (1,3,2) → (1,3,2) (3,2,3)

pop

1. 2인 좌표값을 찾는다

2. 탐색하려는 좌표 작성

3. Q 만드고 만들고 시작점 생성.
(열, 행, 이동거리)

4. while문 (Q의 길이가 없을때까지) 실행

① pop(0) → 앞좌표 나

② 상하좌우 탐색 (pop(0) 기준)
→ 갈 수 있는 지점 Q에 append

5. 3인 지점 찾으면 길이 리턴