S1

미로 탐색 (baekjoon 2178)

categories: 그래프 이론, 그래프 탐색, 너비 우선 탐색

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
1 초	192 MB	147262	64127	41166	42.253%

문제

N×M크기의 배열로 표현되는 미로가 있다.

1	0	1	1	1	1
1	0	1	0	1	0
1	0	1	0	1	1
1	1	1	0	1	1

미로에서 1은 이동할 수 있는 칸을 나타내고, 0은 이동할 수 없는 칸을 나타낸다.
이러한 미로가 주어졌을 때, (1, 1)에서 출발하여 (N, M)의 위치로 이동할 때 지나야 하는 최소의 칸 수를 구하는 프로그램을 작성하시오. 한 칸에서 다른 칸으로 이동할 때, 서로 인접한 칸으로만 이동할 수 있다.
위의 예에서는 15칸을 지나야 (N, M)의 위치로 이동할 수 있다.
칸을 셀 때에는 시작 위치와 도착 위치도 포함한다.

입력

첫째 줄에 두 정수 N, M($2 \le N, M \le 100$)이 주어진다. 다음 N개의 줄에는 M개의 정수로 미로가 주어진다. 각각의 수들은 **붙어서** 입력으로 주어진다.

출력

첫째 줄에 지나야 하는 최소의 칸 수를 출력한다. 항상 도착위치로 이동할 수 있는 경우만 입력으로 주어진다.

문제 출처: BAE/KJOON ONLINE JUDGE
https://www.acmicpc.net/problem/2178

Woody K
Contact: woody35545@gmail.com
Github: https://github.com/woody35545
CNU, Computer Science and Engineering

S1

미로 탐색 (baekjoon 2178)

categories: 그래프 이론, 그래프 탐색, 너비 우선 탐색

입/출력 예시 1

4 6 101111 101010 101011 111011	15

입/출력 예시 2

4 6 110110 110110 111111 111101	9

입/출력 예시 3

2 25 1011101110111011101110111 1110111011101110111011101	38

입/출력 예시 4

7 7 1011111 1110001 1000001 1000001 1000001 1000001 1111111	13

GitHub

Woody K

Contact: woody35545@gmail.com

Github: https://github.com/woody35545

CNU, Computer Science and Engineering

S1

미로 탐색 (baekjoon 2178)

categories: 그래프 이론, 그래프 탐색, 너비 우선 탐색

Answer(1/1)

```python
N, M = 0, 0

def check_valid_pos(x_pos, y_pos):
    if 0 <= x_pos < N and 0 <= y_pos < M:
        return True

    return False


delta = [(-1, 0), (1, 0), (0, -1), (0, 1)]  # up, down, left, right


def bfs(graph, start_node):
    queue = [start_node]
    visited = [[False] * M for _ in range(N)]
    while queue:

        cur = queue.pop(0)

        if not visited[cur[0]][cur[1]]:
            visited[cur[0]][cur[1]] = True

            for k in range(len(delta)):
                next_x = cur[0] + delta[k][0]
                next_y = cur[1] + delta[k][1]

                # check next pos valid
                if check_valid_pos(next_x, next_y):
                    if graph[next_x][next_y] != 0 and not visited[next_x][next_y]:
                        graph[next_x][next_y] = graph[cur[0]][cur[1]] + 1
                        queue.append((next_x, next_y))
    return visited


N, M = map(int, input().split(" "))
graph = []

for _ in range(N):
    graph.append(list(map(int, input())))

bfs(graph, (0, 0))
print(graph[N-1][M-1])

```


GitHub

Woody K

Contact: woody35545@gmail.com

Github: https://github.com/woody35545

CNU, Computer Science and Engineering