



탈출

(baekjoon 3055)

categories: 그래프 이론, 그래프 탐색, 너비 우선 탐색

시간 제한	메모리 제한	제출	정답	맞힌 사람	정답 비율
1 초	128 MB	42779	14571	9885	32.077%

문제

사악한 암흑의 군주 이만혁은 드디어 마법 구슬을 손에 넣었고, 그 능력을 실험해보기 위해 근처의 티뱀숲에 홍수를 일으키려고 한다. 이 숲에는 고슴도치가 한 마리 살고 있다. 고슴도치는 제일 친한 친구인 비버의 굴로 가능한 빨리 도망가 홍수를 피하려고 한다. 티뱀숲의 지도는 R행 C열로 이루어져 있다. 비어있는 곳은 '.'로 표시되어 있고, 물이 차있는 지역은 '*' , 돌은 'X'로 표시되어 있다. 비버의 굴은 'D'로, 고슴도치의 위치는 'S'로 나타내어져 있다.

매 분마다 고슴도치는 현재 있는 칸과 인접한 네 칸 중 하나로 이동할 수 있다. (위, 아래, 오른쪽, 왼쪽). 물도 매 분마다 비어있는 칸으로 확장한다.

물이 있는 칸과 인접해있는 비어있는 칸(적어도 한 변을 공유)은 물이 차게 된다. 물과 고슴도치는 돌을 통과할 수 없다. 또, 고슴도치는 물로 차있는 구역으로 이동할 수 없고, 물도 비버의 소굴로 이동할 수 없다.

티뱀숲의 지도가 주어졌을 때, 고슴도치가 안전하게 비버의 굴로 이동하기 위해 필요한 최소 시간을 구하는 프로그램을 작성하시오.

고슴도치는 물이 찰 예정인 칸으로 이동할 수 없다. 즉, 다음 시간에 물이 찰 예정인 칸으로 고슴도치는 이동할 수 없다. 이동할 수 있으면 고슴도치가 물에 빠지기 때문이다.

입력

첫째 줄에 50보다 작거나 같은 자연수 R과 C가 주어진다.

다음 R개 줄에는 티뱀숲의 지도가 주어지며, 문제에서 설명한 문자만 주어진다. 'D'와 'S'는 하나씩만 주어진다.

출력

첫째 줄에 고슴도치가 비버의 굴로 이동할 수 있는 가장 빠른 시간을 출력한다.

만약, 안전하게 비버의 굴로 이동할 수 없다면, "KAKTUS"를 출력한다.





탈출

(baekjoon 3055)

categories: 그래프 이론, 그래프 탐색, 너비 우선 탐색

예제입력

```
3 3
D . *
...
.S .
```

예제출력

```
3
```

예제입력

```
3 3
D . *
...
..S
```

예제출력

```
KAKTUS
```

예제입력

```
3 6
D...*.
.X.X..
....S.
```

예제출력

```
6
```

예제입력

```
5 4
.D.*
....
..X.
S.*.
....
```

예제출력

```
4
```



탈출

(baekjoon 3055)

categories: 그래프 이론, 그래프 탐색, 너비 우선 탐색

Answer(1/2)

```
1 # Notations
2 TARGET = 'D' # Destination
3 PLAYER = 'S' # Player
4 WATER = '*'
5 ROCK = 'X'
6
7 PLAYER_CANT_MOVE = [ROCK, WATER]
8 WATER_CANT_EXTEND = [ROCK, TARGET, WATER]
9 world_map = []
10 R, C = 0, 0
11
12 dir = [(0, 1), (0, -1), (1, 0), (-1, 0)]
13
14
15 def extend_water():
16     global world_map
17
18     newly_added = []
19     for i in range(R):
20         for j in range(C):
21             if world_map[i][j] == WATER:
22                 for move in dir:
23                     nx = i + move[0]
24                     ny = j + move[1]
25
26                     # 물을 확장하려는 좌표가 유효한 범위인지 확인하고 물이 확장될 수 있는 영역인지 확인
27                     if 0 <= nx < R and 0 <= ny < C:
28                         if world_map[nx][ny] not in WATER_CANT_EXTEND:
29                             if (i, j) not in newly_added:
30                                 world_map[nx][ny] = WATER
31                                 newly_added.append((nx, ny))
32
33
```



Woody K

Contact: woody35545@gmail.com

Github: <https://github.com/woody35545>

CNU, Computer Science and Engineering



탈출

(baekjoon 3055)

categories: 그래프 이론, 그래프 탐색, 너비 우선 탐색

Answer(2/2)

```

34 def bfs(graph, start_node:tuple, dest_node):
35     queue = [start_node]
36     visited = [[False]*C for _ in range(R)]
37     visited[start_node[0]][start_node[1]] = True
38     cost = [[0]*C for _ in range(R)]
39
40     while queue:
41
42         extend_water() # 물을 확장시켜놓고 시작하면 다음 확장지역에 가지 못하게 하는 조건을 짜기 수월하므로 확장시키고 시작
43
44         for _ in range(len(queue)):
45             # 지금 초기화 되어있는 Queue의 원소들을 모두 처리한게 한 라운드로 해야한다.
46             cx, cy = queue.pop(0)
47
48             for diff in dir:
49                 nx, ny = cx + diff[0], cy + diff[1]
50
51                 # check next position is valid
52                 if 0 <= nx < R and 0 <= ny < C and graph[nx][ny] not in PLAYER_CANT_MOVE and not visited[nx][ny]:
53                     queue.append((nx, ny))
54                     visited[nx][ny] = True
55                     cost[nx][ny] = cost[cx][cy] + 1
56
57
58             if cost[dest_node[0]][dest_node[1]] == 0:
59                 print("KAKTUS")
60             else:
61                 print(cost[dest_node[0]][dest_node[1]])
62
63     return cost
64
65 # init world map
66 R, C = map(int, input().split(" "))
67 for i in range(R):
68     world_map.append(list(input()))
69
70 for i in range(R):
71     for j in range(C):
72         if world_map[i][j] == "PLAYER":
73             start_position = (i, j)
74         if world_map[i][j] == "TARGET":
75             destination_position = (i, j)
76
77 # bfs start
78 cost = bfs(world_map, start_position, destination_position)

```



Woody K

Contact: woody35545@gmail.com

Github: <https://github.com/woody35545>

CNU, Computer Science and Engineering