

PREPARE^{NEW}

CERTIFY

COMPETE

Search



C_24060121140112 ▾

[All Contests](#) > [Praktikum ASA Undip pertemuan 4](#) > [Labirin](#)

Labirin

Problem

Submissions

Leaderboard

Discussions

Author : Albar N

Kamu diberi peta labirin, dan tugas kamu adalah menemukan jalur dari awal hingga akhir. Kamu bisa berjalan ke kiri(L), kanan(R), atas(U) dan bawah(D). Jalur yang dapat dilewati adalah '.'. Sedangkan '#' adalah dinding yang tidak bisa dilalui.

Input Format

- Baris input pertama memiliki dua bilangan bulat n dan m: tinggi dan lebar peta.
- Lalu ada n baris m karakter yang menggambarkan labirin. Setiap karakter adalah . (lantai), # (dinding), A (mulai), dan B (akhir). Tepatnya ada satu A dan satu B di input.

Constraints

 $1 < n, m < 1000$

Output Format

- Pertama cetak "Ya", jika ada jalur, dan "Tidak" jika tidak.
- Jika ada jalur, cetak panjang jalur terpendek tersebut dan deskripsinya sebagai string yang terdiri dari karakter L (kiri), R (kanan), U (atas), dan D (bawah).

Sample Input 0

```
5 8
#####
#.A#...#
#.#.#B#
#.....#
#####
```

Sample Output 0

```
Ya
9
LDDRRRRRU
```

[f](#) [t](#) [in](#)Contest ends in 15 days

Submissions: 2

Max Score: 100

Rate This Challenge:

☆☆☆☆☆

[More](#)

Python 3 ▾



```
1 from collections import deque
```

```
2
3 n, m = map(int, input().split())
4 maze = [list(input()) for _ in range(n)]
5
6 start = None
7 end = None
8 for i in range(n):
9     for j in range(m):
10         if maze[i][j] == 'A':
11             start = (i, j)
12         elif maze[i][j] == 'B':
13             end = (i, j)
14
15 def bfs(maze, start, end):
16     queue = deque([start])
17     visited = set()
18     directions = {'D': (1, 0), 'U': (-1, 0), 'R': (0, 1), 'L': (0, -1)}
19     came_from = {}
20
21     while queue:
22         current = queue.popleft()
23         if current == end:
24             return came_from
25
26         for direction, (dx, dy) in directions.items():
27             next_pos = (current[0] + dx, current[1] + dy)
28             if (
29                 0 <= next_pos[0] < n and
30                 0 <= next_pos[1] < m and
31                 maze[next_pos[0]][next_pos[1]] != '#' and
32                 next_pos not in visited
33             ):
34                 queue.append(next_pos)
35                 visited.add(next_pos)
36                 came_from[next_pos] = (current, direction)
37
38     return None
39
40 def reconstruct_path(came_from, start, end):
41     current = end
42     path = []
43     while current != start:
44         current, direction = came_from[current]
45         path.append(direction)
46     path.reverse()
47     return path
48
49 came_from = bfs(maze, start, end)
50 if came_from is None:
51     print('Tidak')
52 else:
53     path = reconstruct_path(came_from, start, end)
54     print('Ya')
55     print(len(path))
56     print(''.join(path))
```

Line: 1 Col: 1

 Upload Code as File ☐ Test against custom input

Run Code

Submit Code