Ice Cave

Big-O Blue – Lecture 5: BFS

Tóm tắt đề bài

Tóm tắt đề bài

- Cho ma trận m x n, mỗi ô có 2 loại:
  - + ‘X’: băng bị nứt
  - + ‘.’ : băng còn nguyên
- Nếu bước vào ô ‘.’ còn nguyên, ô đó sẽ bị nứt → ‘X’
- Nếu bước vào ô ‘X’ → bạn sẽ bị rơi xuống

Yêu cầu: Bắt đầu từ ô (r1, c1). Hãy kiểm tra xem bạn có thể rơi xuống ô (r2, c2) không

Input - Output

Input

- Dòng đầu tiên chứa hai số nguyên *n, m* là số hàng và số cột (1 ≤ n, m ≤ 500)
- n dòng tiếp theo là ma trận trò chơi gồm các ô chỉ có 2 giá trị ‘X’ hoặc ‘.’
- Dòng tiếp theo là 2 số r1, c1
- Dòng cuối cùng là 2 số r2, c2

Output

- In ra “YES” nếu có thể xuất phát từ ô (r1, c1) và rơi xuống ô (r2, c2) ngược lại in ra “NO”

Ví dụ

Ví dụ 1

Input

```
4 6
X...XX
...XX.
.X..X.
......
1 6
2 2
```

- Ta cần xuất phát ở ô (1, 6) và rơi xuống ở ô (2, 2)
- Thứ tự đi: (1, 6) → (2, 6) → (3, 6) → (4, 6) → (4, 5) → (4, 4) → (3, 4) → (3, 3) → (2, 3) → (2, 2) → (2, 1) → (2, 2)
- Lần đầu tiên bước vào ô (2, 2) đã làm nó bị nứt
- Lần thứ 2 bước vào (2, 2) thì người chơi sẽ bị rơi xuống

Output

```
YES
```

Ví dụ 2

Input

```
5 4
.X..
...X
X.X.
....
.XX.
5 3
1 1
```

- Ta xuất phát từ ô (5, 3) và muốn rơi ở ô (1, 1)
- Thứ tự đi: (5, 3) → (4, 3) → (4, 2) → (3, 2) → (2, 2) → (2, 1) → (1, 1)
- Lần đầu tiên bước vào ô (1, 1) làm nó bị nứt. Nhưng ta không thể bước vào ô nào khác nữa

Output

```
NO
```

Ví dụ 3

Input

```
4 7
..X.XX.
.XX..X.
X...X..
X......
2 2
1 6
```

- Ta xuất phát tại ô (2, 2) và muốn rơi ở ô (1, 6)
- Thứ tự đi: (2, 2) → (3, 2) → (4, 2) → (4, 3) → (4, 4) → (4, 5) → (4, 6) → (4, 7) → (3, 7) → (2, 7) → (1, 7) → (1, 6)
- Tại ô (1, 6) đã nứt sẵn nên khi đến ô này, người chơi sẽ lập tức rơi xuống

Output

```
YES
```

Phân tích bài toán

Phân tích bài toán

- Bài toán cần tìm 1 đường đi từ ô (r1, c1) và rơi xuống ở ô (r2, c2), không được phép bước vào 1 ô ‘X’ nào khác ô (r2, c2) trên đường đi.
- Có 2 trường hợp:
  - Không thể đi từ (r1, c1) tới (r2, c2) → NO
  - Có thể đi từ (r1, c1) tới (r2, c2):
    + Có thể rơi xuống ô (r2, c2) → YES
    + Không thể rơi xuống ô (r2, c2) → NO

Phân tích bài toán

- Trường hợp 1: Sử dụng BFS nhưng không thể thăm tới (r_2, c_2) từ (r_1, c_1).

- Trường hợp 2:
  - 2.1: Nếu ô (r_2, c_2) sẵn là ô 'X', chỉ cần nhảy vào và rơi xuống, nếu nó đang là '.', cố gắng biến nó thành ô 'X', và nhảy vào 1 lần nữa.

  - 2.2: Nếu ô (r_2, c_2) đang là ô '.', biến nó thành ô 'X', nhưng không có cách nào để nhảy vào lại ô (r_2, c_2).

Phân tích bài toán

- Để giải quyết trường hợp 2.1, thực hiện duyệt BFS, tìm 1 đường đi từ (r_1, c_1) tới (r_2, c_2):
  - Chỉ thăm tới những ô '.' (ngoại trừ ô (r_2, c_2)).
  - Nếu ô (r_2, c_2) đang là '.', thăm nó và đánh dấu lại là ô 'X'.
  - Nếu thăm vào ô (r_2, c_2) khi nó đang là 'X' → YES.

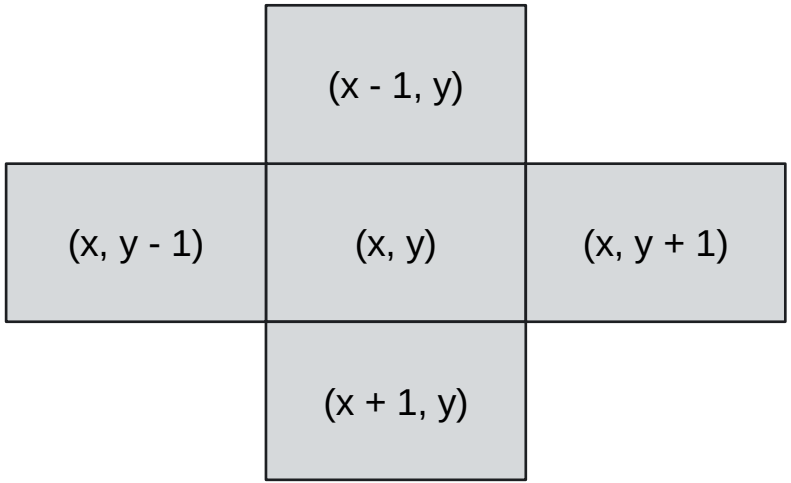
- Nếu không rơi vào trường hợp 2.1 → NO.

Phân tích bài toán

Các ô kề cạnh nhau:

+ Nếu coi (x, y) là 1 đỉnh
+ Các đỉnh kề với (x, y) là:
  (x, y + 1) → (x, y) + (0, 1)
  (x, y – 1) → (x, y) + (0, -1)
  (x + 1, y) → (x, y) + (1, 0)
  (x – 1, y) → (x, y) + (-1, 0)

→ Ta có mảng các hướng đi:
dr = [0, 0, 1, -1]
dc = [1, -1, 0, 0]



Các bước thuật toán

Các bước thuật toán

- Bước 1: Nhập dữ liệu
- Bước 2: BFS từ ô (r_1, c_1), chỉ thăm tới những ô '.', ngoại trừ ô (r_2, c_2).
- Bước 3: Kết luận:
  - Nếu thăm được ô (r_2, c_2) khi nó đang 'X' → YES
  - Ngược lại → NO

Độ Phức tạp:
Time: O(M*N)
Space: O(M*N)

Mã giả

Mã giả

```
read (n, m)
read(a)
read(r1, c1)
read(r2, c2)
if BFS(r1-1, c1-1, r2-1, c2-1) == True:
    print("YES")
else:
    print("NO")
```

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

function BFS(r1, c1, r2, c2):
    queue q
    q.push((r1, c1))
    a[r1][c1] = 'X'
    while q not empty:
        ur, uc = q.pop()
        for i = 0 to 3:
            x = ur + dr[i]
            y = uc + dc[i]
            if x == r2 and y == c2 and a[x][y] == 'X':
                return True
            if x, y in matrix and a[x][y] == '.':
                q.push((x, y))
                a[x][y] = 'X'
    return False
```