## Bombs! NO they are Mines!!

### Input:

R, C <= 1000,

Range (0, 0) -> (R-1, C-1)

### Output:

Shortest path from

### Sample testcase:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 0 |  |  |  |  | v2 |  |  |  |  |  |
| 1 | x |  |  | v1 | U | v3 |  |  |  |  |
| 2 | x |  |  |  | v4 |  |  |  |  |  |
| 3 | x |  |  |  |  |  |  |  |  |  |
| 4 | x | x | x | x | x | y |  |  |  |  |
| 5 |  |  |  |  | x | y |  |  |  |  |
| 6 |  |  |  |  | x | y |  |  |  |  |
| 7 |  |  |  |  | x | y | y |  |  |  |
| 8 |  |  |  |  | x | x | y |  |  |  |
| 9 |  |  |  |  |  | x | x | x | x |  |

### Ý tưởng



BFS: không có trọng số

Dijkstra: có trọng số không âm

=> Sử dụng thuật toán BFS,

mảng 2 chiều, đánh dấu trạng thái của ô, (visited)

### Code:

dr = {0, 0, -1, 1}

dc = {-1, 1, 0, 0}

function BFS(int sx, int sy) {

int dist[1001][1001];

for(i: 0 -> R-1) {

for(j; 0 -> C-1) {

dist[i][j] = INF

}

}

dist[sx][sy] = 0

queue <[int, int]> qu

while(!qu.empty()) {

[ux, uy] = qu.top()

qu.pop()

for(i: 0 -> 3) {

vx, vy

vx = ux + dr[i]

vy = uy + dr[i]

if (vx < 0 || vx >= r || vy < 0 || vy > c) {

continue;

}

if (dist[vx][vy] > dist[ux][uy] + 1 && bom[vx][vy] != -1) {

dist[vx][vy] + dist[ux][uy] + 1

qu.add([vx, vy])

}

}

}

}

function main() {

while(true) {

input(R, C);

if(R == 0 && C == 0) {

end

}

for(i: 0 -> R-1) {

for(j: 0 -> C-1) {

bom[i][j] = 0

}

}

input(rows)

for(i : 1 -> rows) {

input(idRow)

input(numBoms)

for(j : 1 -> numBoms) {

input(idCol)

bom[idRow][idCol] = -1;

}

}

input(sx, sy)

input(tx, ty)

BFS(sx, sy)

output(dist[tx][ty])

}

}

Độ phức tapj: O(V + E) = O(R \* C \* số test)