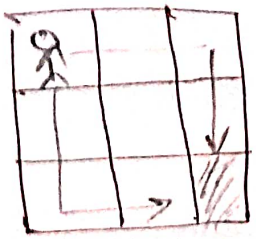
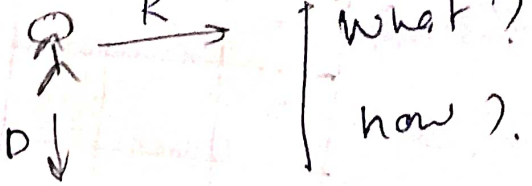
Backtracking

e.g





→ R R D D

- D D R R

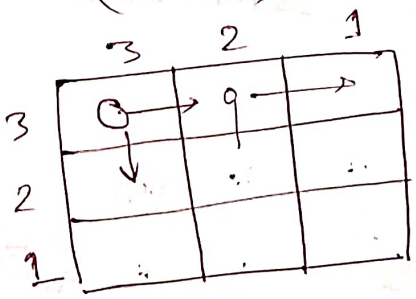
- R D D R. / R D D R many ways to get to gray scale.

(0, 0) ⟶ (2, 2).

many ways.

Lets pass the matrix.

(3, 3)



r c

(3, 3)

L R

(2, 3) (3)

(3, 2) (3)

(1) (3, 1)

(1, 3)

(2, 3) [2 paths]

(2, 3)

Ans (1)

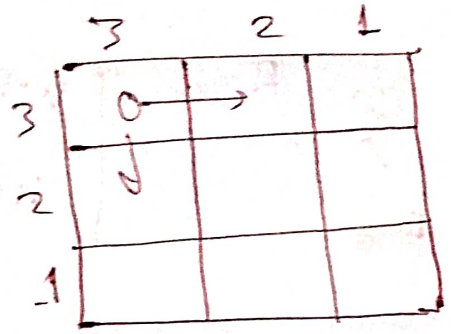
(2, 1)

(1 path)

(2, 2)

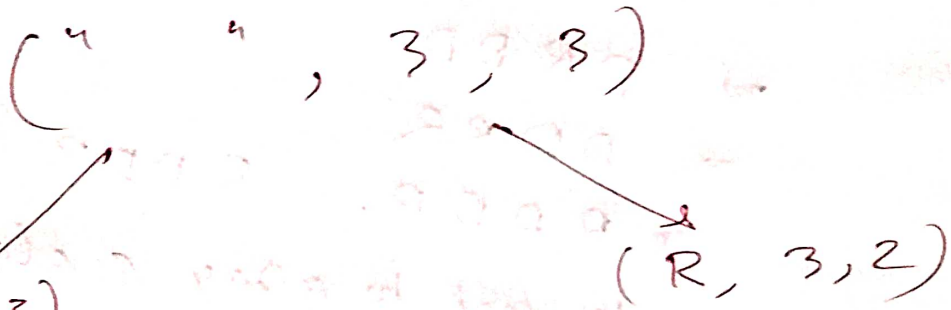
(1 path)

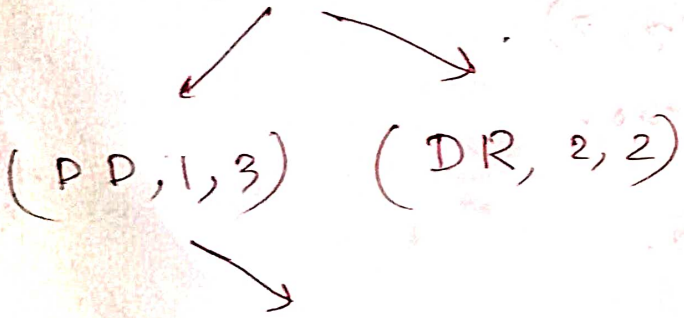
Maze problem



→ R

↓ D

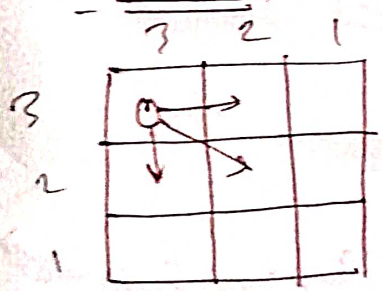


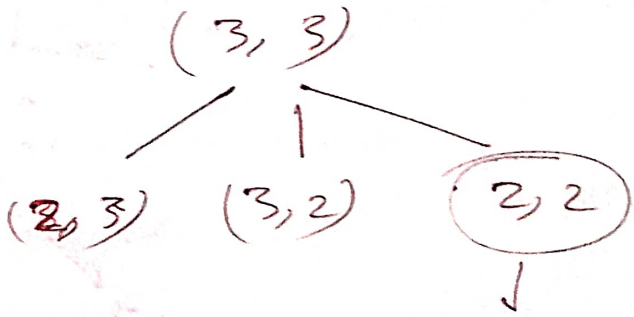


→ (D D R R , 1 , 1) — base case

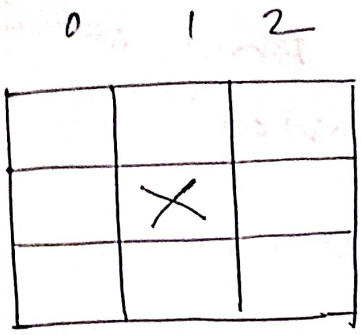
m

Diagonally

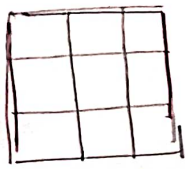




9/ Maze with obstacles.

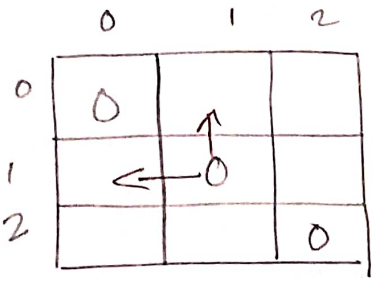


→ Here comes the concept of Diagonal movement.



maze[0].length
= ~~no of colum~~
= length of colum.

(8)



→ All path.

→ stack overflow problem.

("", 0, 0)

(R, 0, 1)

(D, 1, 0)

(DD, 2, 0)

(DR, 1, 1)

(DU 0, 0)

problem.
(it crossed to the same location, from where we all started)

→ how to solve this problem.

mark the cell, that we have visited, prevent it from moving to the cell which cause the problem.