Backtracking

↳ exploring all the path

(i) N Queen Problem.

(ii) Rate in a maze

(iii) subset sum

(IV) Suduko Solver

N- Queen Problem

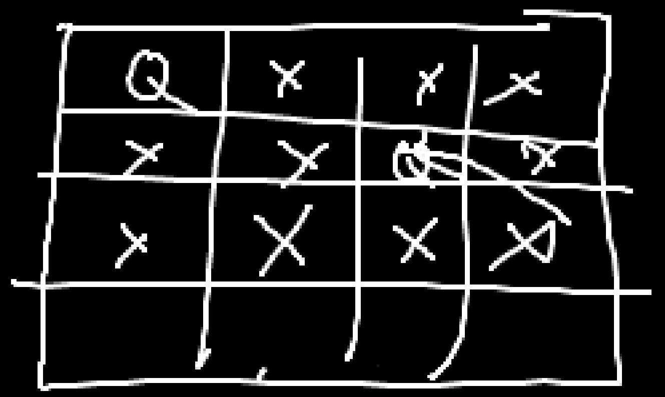
eg· 4 X4 matrix

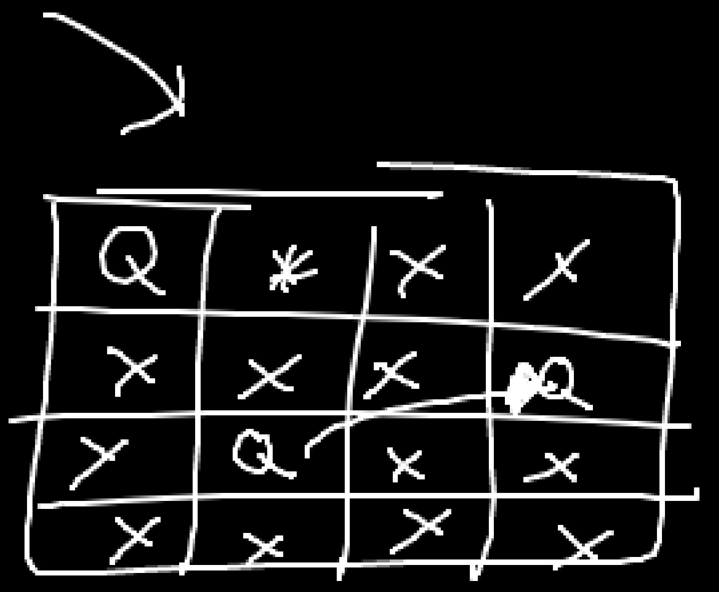
N X N matrics

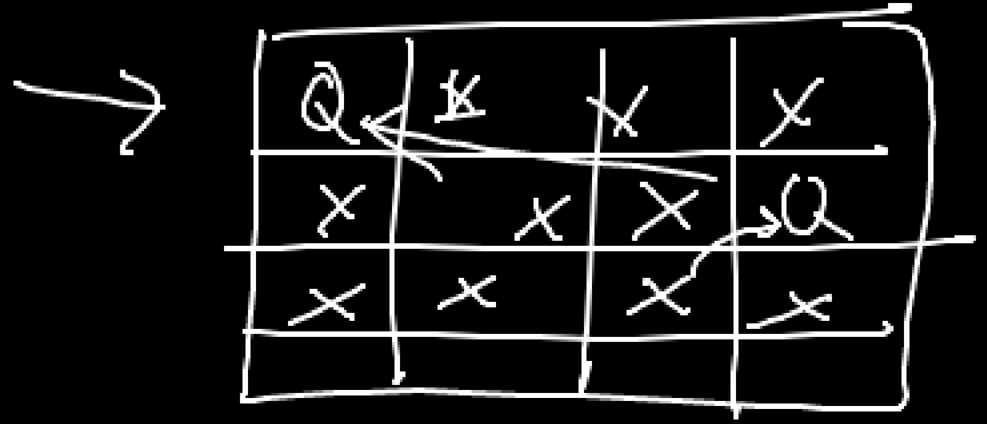
(I) Not in same row ←

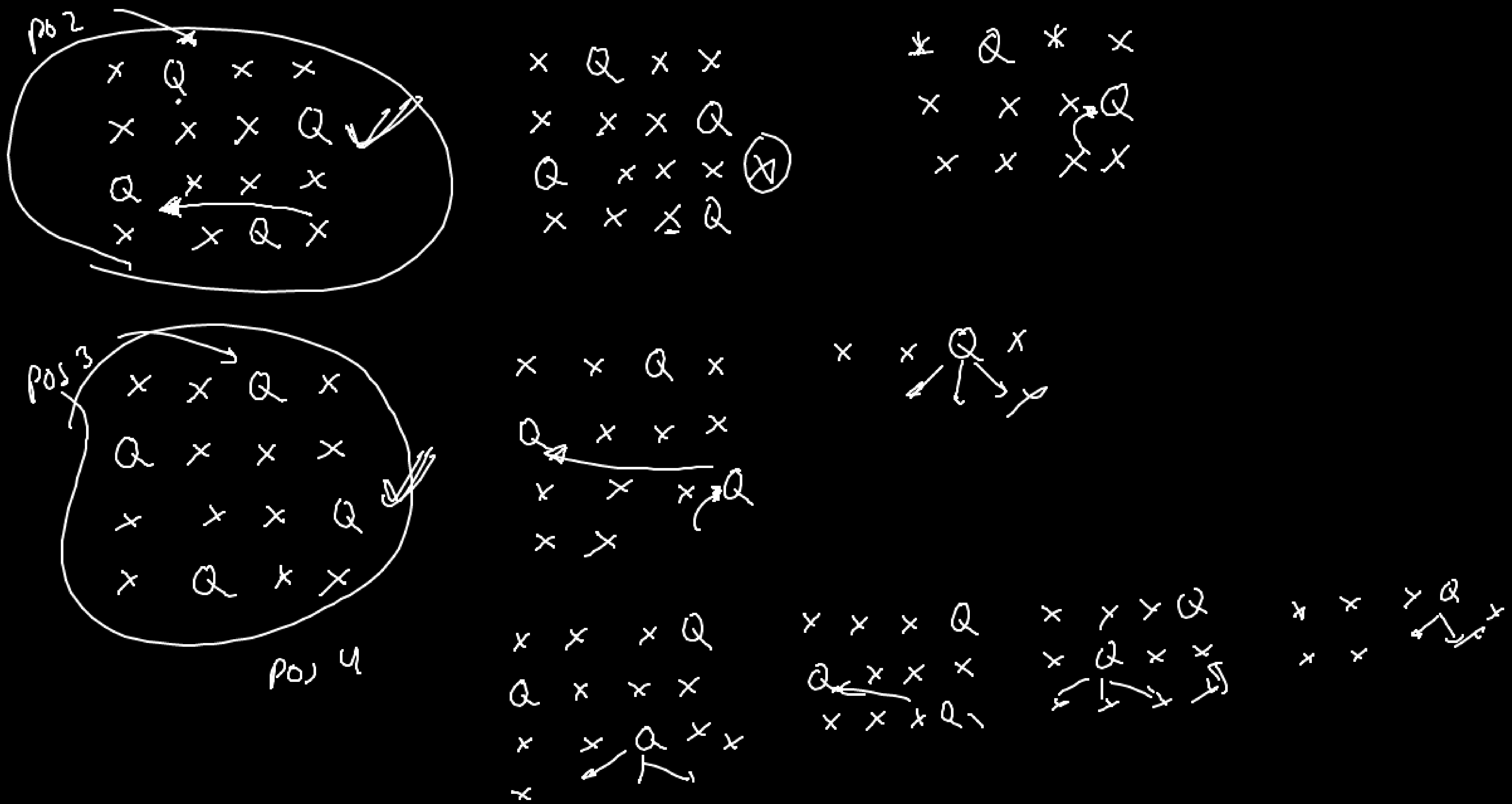
(II) Not in same col ↑

(III) Not in some diagonal.









pos 2
pos 3
pos 4

x	Q	x	x
x	x	x	Q
Q	x	x	x
x	x	Q	x

x	x	Q	x
Q	x	x	x
x	x	x	Q
x	Q	x	x

✓

for N = 4

→ Two possible path

① Not in same col

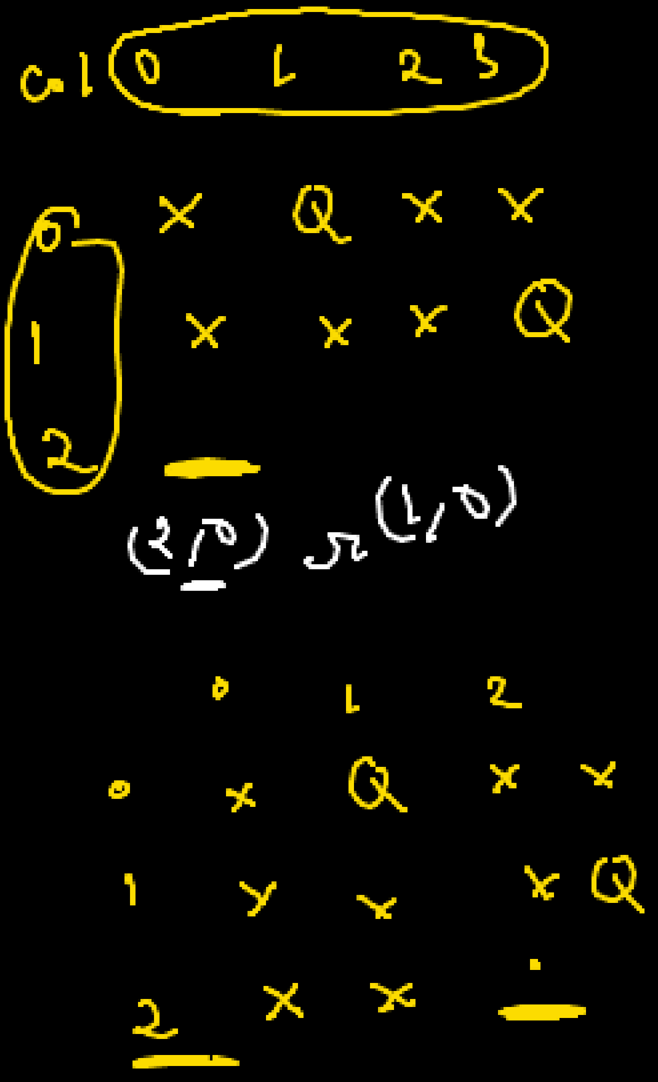
(ii) ———→ row

(iii) ———→ dia

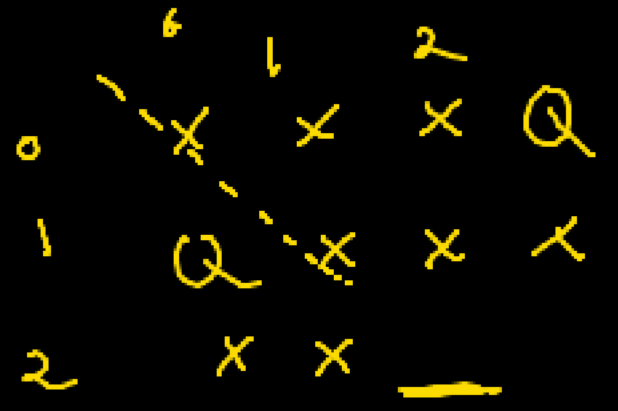
} Cond^n for placing Queen

```
bool isPossible (int n, int row, int col){
    //checking for same col
    for (int i=row-1; i>=0; i--){
        if (board [i][col]==1)
            return false;
    }

    //checking for same row;
    for (int j= col-1; j>=0; j--)
    {       if ( board[row][j] ==1)
                return false;
    }
```

This is optional

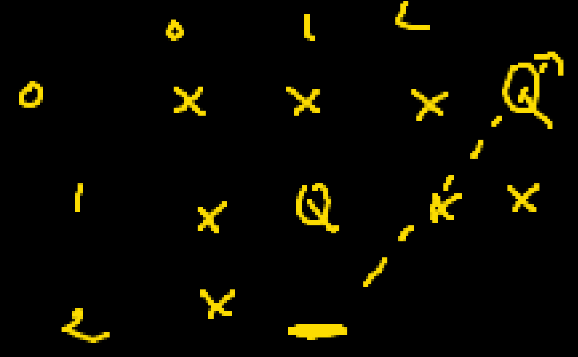


```
//checking for left diagonal
for (int i = row-1, j = col-1; i>=0 && j>=0; i--, j--){
{   if (board[i][j] == 1)
            return false;

}
```



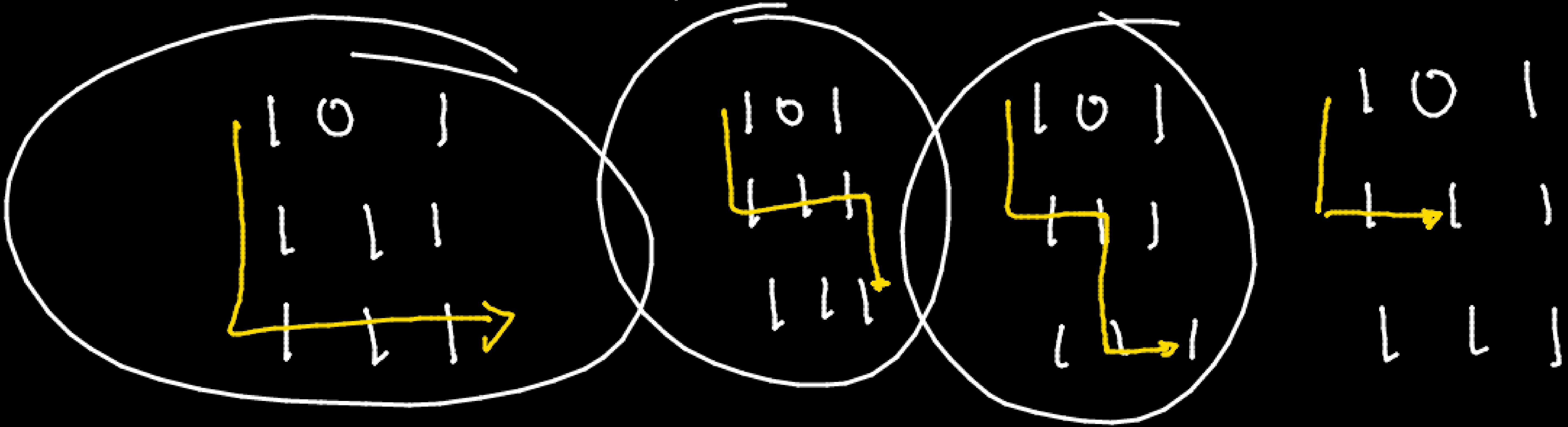
(2,2)
↓
check (1,1)

```
// checking for right diagonal
for (int i = row-1, j = col+1; i>=0 && j<n; i--,j++)
{    if board[i][j] == 1;
            return false;

 }
 return true;
}
```



(2,1)
↓
check (1,2)

Rate in Maze



1 0 1
1 1 1
1 1 1

1 0 1
1 1 1
1 1 1

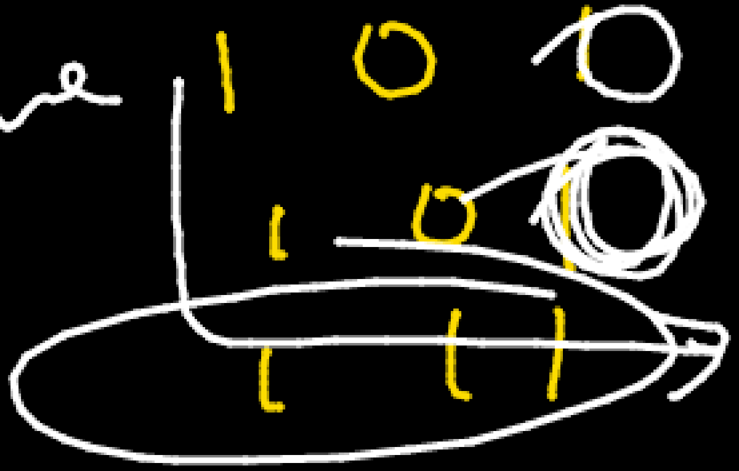
1 0 1
1 1 1
1 1 1

1 0 1
1 1 1
1 1 1

Condn

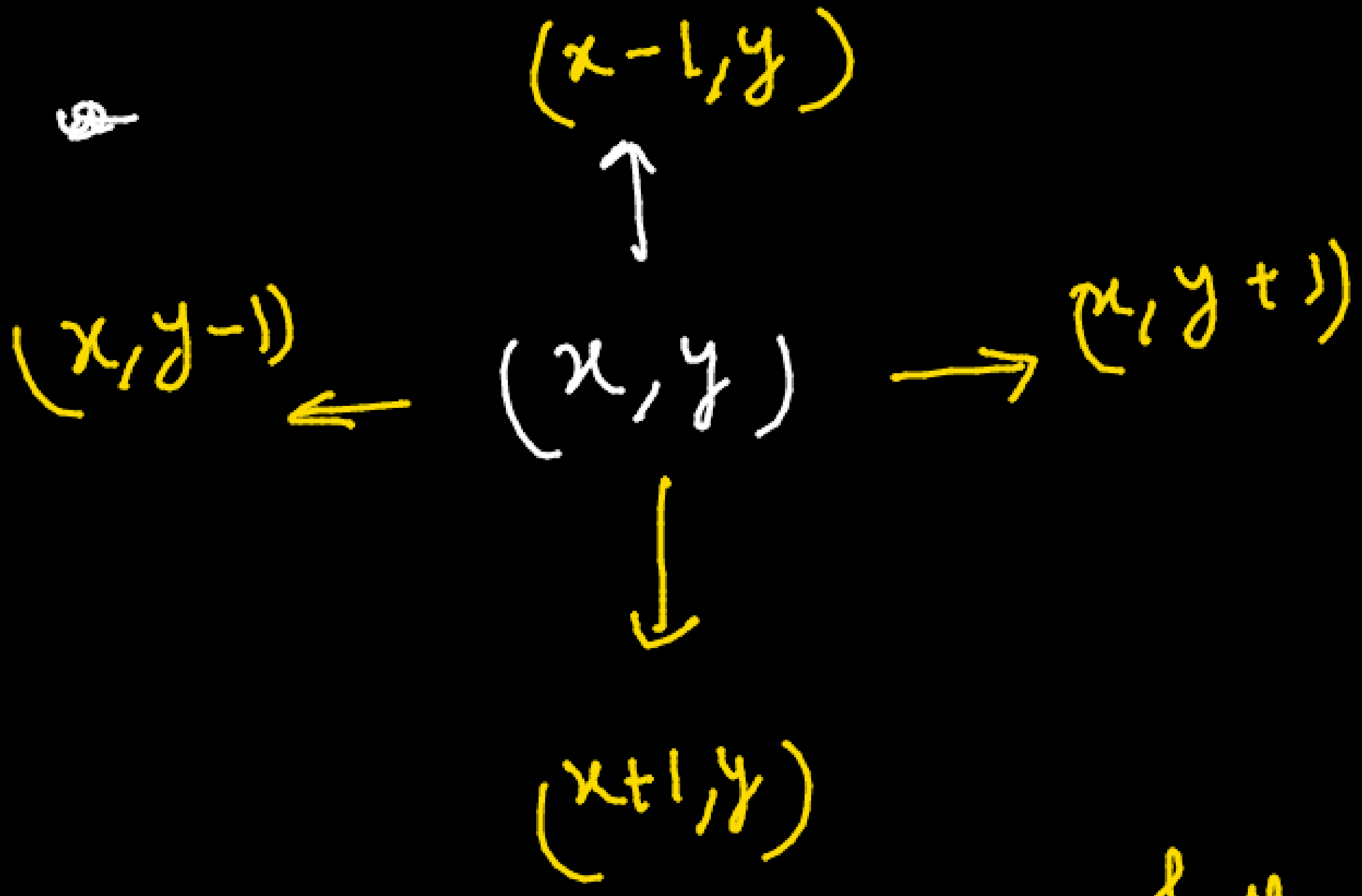
0 → You can't move

1 → you can move

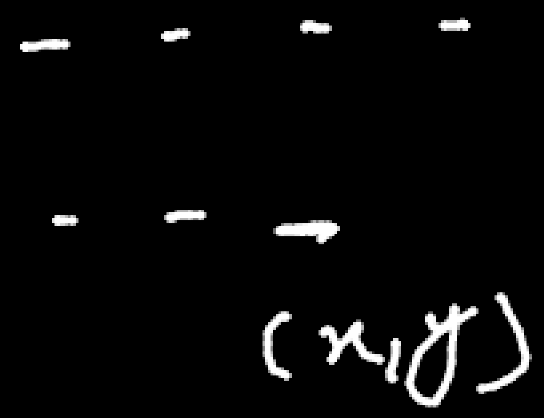


1 0 0
1 0 0
1 1 1

1 cell one time



up, down, left right



foue conditions

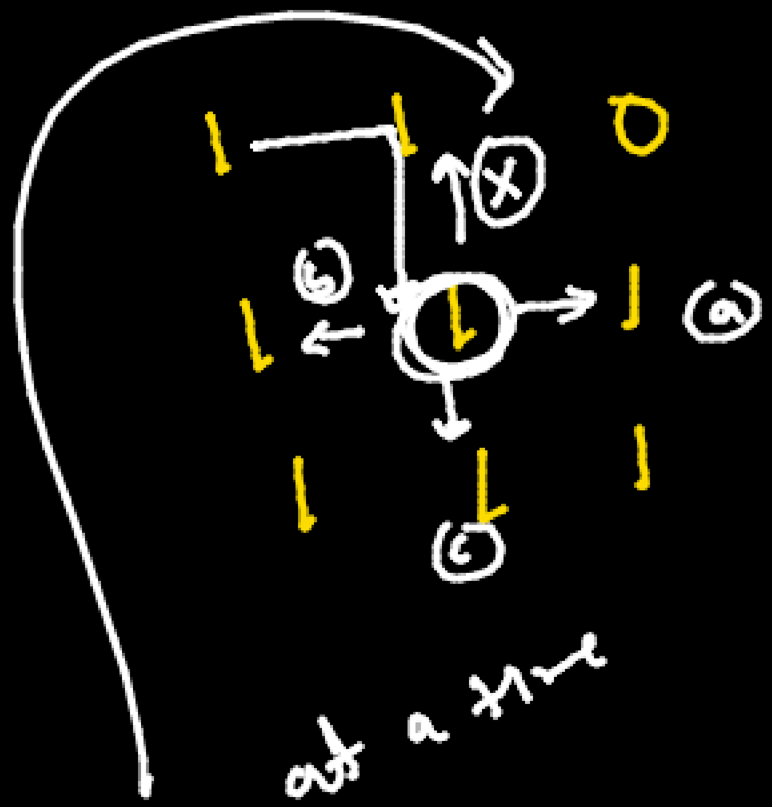
solution [18][18]   board [18][18]

```
void ratHelper (int n, int x, int y) {
  if (x==n-1 && y ==n-1)
  {        solution [x][y] = 1;
              print (n);
           solution [x][y] =0;
           return;
  }
   //condn for moving out of maze/board.
   if ( x >= n || y >= n || x < 0 || y < 0 || board[x][y]==0
              || solution [x][y] ==1)
   {
      return;
   }
```

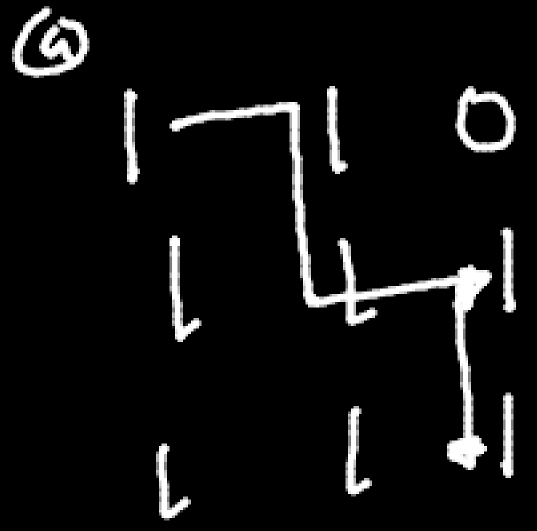
$$\begin{array}{c|ccc} & 0 & 1 & 2 \\ \hline 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 2 & 1 & 1 & 1 \end{array}$$

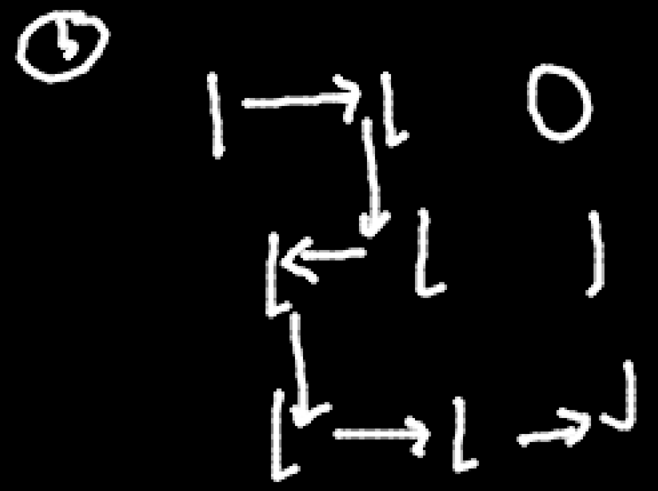
```
solution [x][y]=1;
ratHelper (x-1, y); //up
ratHelper (x+1, y)  //down
ratHelper (x , y-1 ); //left
ratHelper (x ,y+1); //right
solution [x][y] = 0;

}
```

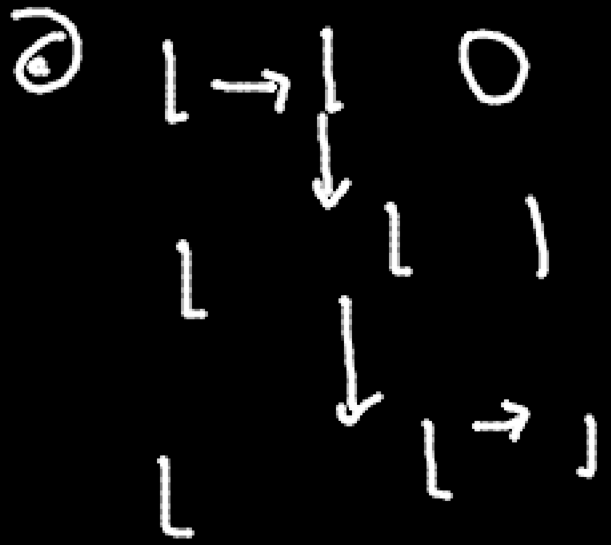
eg.



Include one at a time cell







u, d, l, r

1 → cell

on time

path

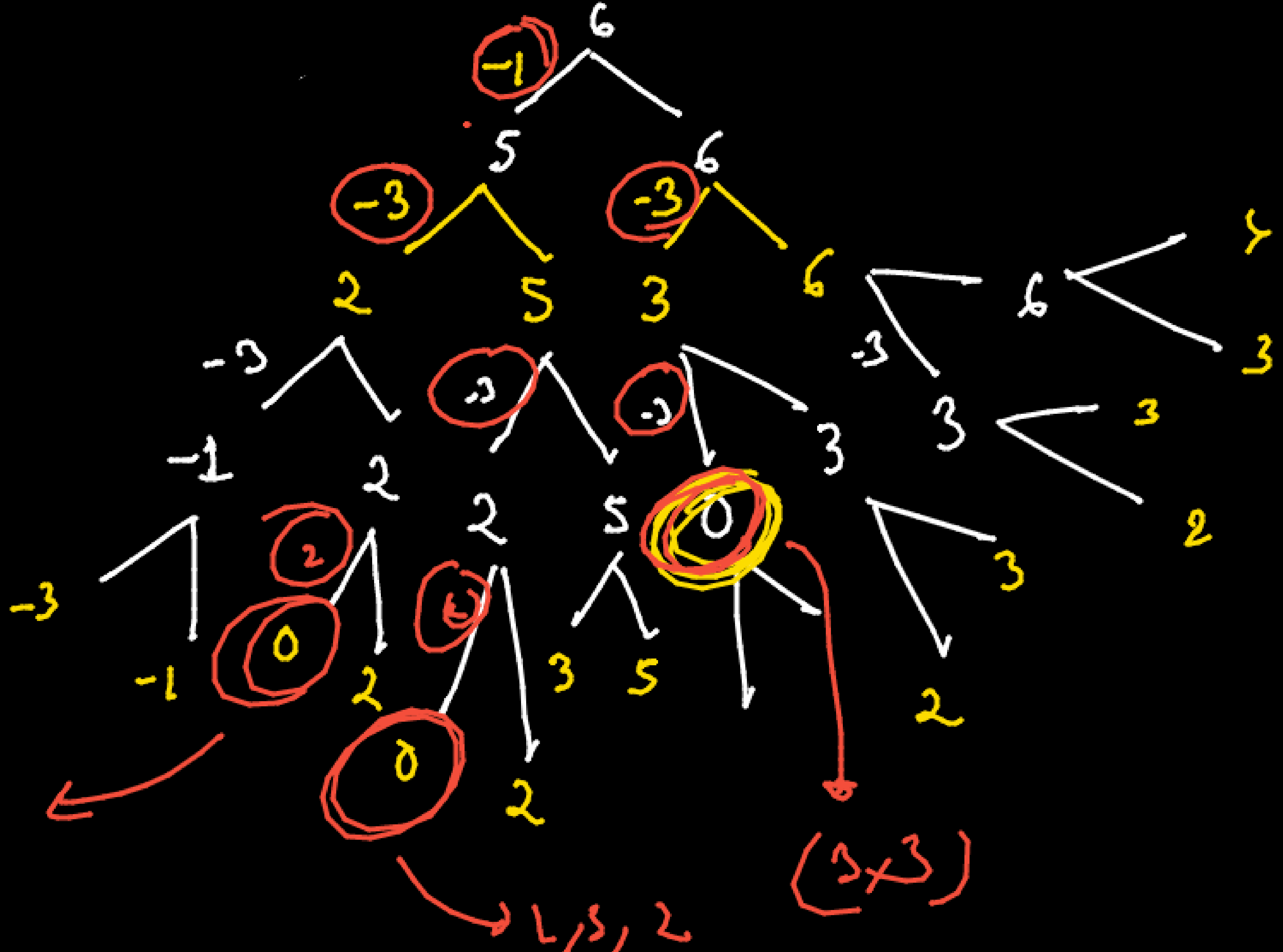
6

1, 3, 3, 2

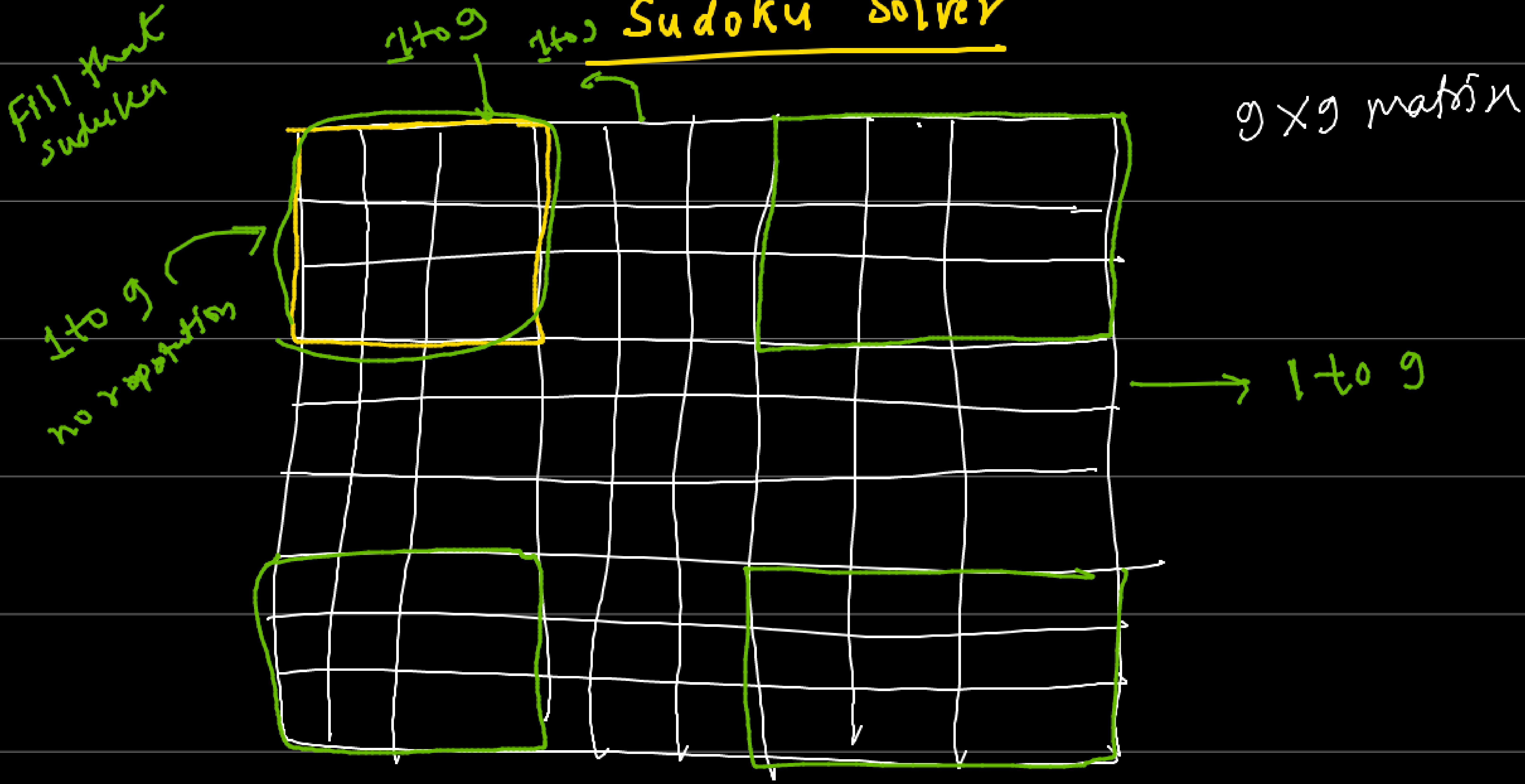
(1, 3, 2)
(1, 3, 2)
(3 3)

2

Subset sum



Sudoku Solver



where 0 no. is there we have to fill no. from (1 to 9)

```
solve_sudko (board){
  if
    findEmptyPos( )
  else
    return true;
  1 to 9  check condn
  fill that no. in empty pos
      solveSudko
}
```

find no. which is not present in row, col and also 3 x 3 matrix

(i) Not present in same row (horizontal)

(ii) Not present in same col (vertical)

(iii) Not present in 3 x 3 matrix (box)

```
solveSudoku ( board) {

(1) Find empty pos of board
(2) If not find, return true;
(3) If, find, start explore that pos from
        1 to 9
          ↳ check   row
          ↳ ———     col
          ↳         box

      board[emptypos] = no.

      solveSudoko(board);

(4) Return false;
```

If return true, return true
If return false, mark pos
empty again and try to fill
will further no.