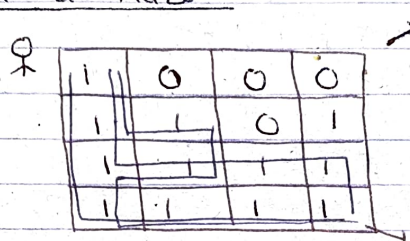
Date: / /
Page:

Day - 95

Recursion - 16

* Rat in a Maze:



⇒ A rat want to go to goal.

⇒ But he can only traverse through 1.

	0	1	2	3
0	∅1	0	0	0
1	∅1	∅1	0	0
2	∅1	∅1	0	0
3	∅1	∅1	∅1	∅1

⇒ Path: D D D R R R

⇒ D D R R R D

D R D D R R

⇒ Total 14 Paths or answers.

Conditions

⇒ You can't go outside the matrix.

⇒ You can't go on 0 cell.

⇒ You can't revisit the visited cell.

Path: D R D L D R R R

⇒ Now, we go back & traverse other paths & then check.

⇒ When we go back, we will revert all the changes.

Up : i-1, j (-1,0) Goal

Down: i+1, j (1,0) (i = n-1, j = n-1)

Left : i, j-1 (0,-1)

Right: i, j+1 (0,1)

```
int row[] = {-1,1,0,0};
int col[] = {0,0,-1,1};
string dir = "UDLR";

void totall(vector<vector<int>> &matrix, int i, int j,
    int n, string path, vector<string>
    &ans, vector<vector<int>> &visited){
    if (i==n-1 && j==n-1){
        ans.push_back(path);
        return;
    }
    visited[i][j] = 1;
    for(int k=0; k<4; k++){
        if (valid(i+row[k], j+col[k], n) &&
            matrix[i+row[k]][j+col[k]] &&
            !visited[i+row[k]][j+col[k]]){
            path.push_back(dir[k]);
            totall(matrix, i+row[k],
        j+col[k], n, path, ans, visited);
            path.pop_back();
        }
    }
    visited[i][j] = 0;
}

bool valid(int i, int j, int n){
    return i>=0 && j>=0 && i<n && j<n;
}
```

Time Complexity

=> All have three paths to go —

So, $3 \times 3 \times \ldots$ (N^2 times) $= 3^{N^2}$

$N \times N = N^2$