Rat in a Maze Problem - I

```
class Solution:
    def findPath(self, m, n):
        # code here
        visited = [[False]*n for i in range(n)]
        psf = ''
        res = []
        self.dfs(visited,m,n,0,0,psf,res)
        return res


    def dfs(self,visited,grid,N,r,c,psf,res):
        if r<0 or c<0 or r>=N or c>=N or visited[r][c]==True or grid[r]
[c]==0:
            return
        if r==N-1 and c==N-1:
            res.append(psf)
            return
        visited[r][c]=True
        self.dfs(visited,grid,N,r+1,c,psf+'D',res)
        self.dfs(visited,grid,N,r,c-1,psf+'L',res)
        self.dfs(visited,grid,N,r,c+1,psf+'R',res)
        self.dfs(visited,grid,N,r-1,c,psf+'U',res)
        visited[r][c]=False
```

Consider a rat placed at (0, 0) in a square matrixof order N * N. It has to reach the destination at (N - 1, N - 1). Find all possible paths that the rat can take to reach from source to destination. The directions in which the rat can move are 'U'(up), 'D'(down), 'L' (left), 'R' (right). Value 0 at a cell in the matrix represents that it is blocked and rat cannot move to it while value 1 at a cell in the matrix represents that rat can be travel through it.
Note: In a path, no cell can be visited more than one time.

Example 1:Input:

```
N = 4
m[][] = {{1, 0, 0, 0},
         {1, 1, 0, 1},
         {1, 1, 0, 0},
         {0, 1, 1, 1}}
Output:
```

```
DDRDRR DRDDRR
Explanation:
The rat can reach the destination at
(3, 3) from (0, 0) by two paths - DRDDRR
and DDRDRR, when printed in sorted order
we get DDRDRR DRDDRR.
```

Example 2:

```
Input:
N = 2
m[][] = {{1, 0},
         {1, 0}}
Output:
-1
Explanation:
No path exists and destination cell is
blocked.
```