Rat in a maze

Approach :

We try all possible ways if a cell 1 is found . We explore all 4 direction and if a path is possible in any direction then we call it recursively for all 4 direction a loop is used and for direction cell we use two array drow and dcol.

```
#include <bits/stdc++.h>
void helper(int row,int col,int n,vector<vector<int>>& maze,vector<vector<int>> temp,vector<vector<int>>& ans)
{
  if(row==n-1 && col==n-1)
  {
      vector<int> res;

      for(int i=0;i<n;i++)
      {
        for(int j=0;j<n;j++)
        {
          if((i==0 && j==0) ||(i==n-1 && j==n-1))
            res.push_back(1);
          else
            res.push_back(temp[i][j]);
        }
      }
      ans.push_back(res);
      return ;
  }
  int drow[]={-1,1,0,0};
  int dcol[]={0,0,-1,1};
  for(int i=0;i<4;i++)
  {
     int nrow=row+drow[i];
     int ncol=col+dcol[i];
     if(nrow>=0 && ncol>=0 && nrow<n && ncol<n && maze[nrow][ncol]==1 &&temp[nrow][ncol]!=1)
     {
        temp[row][col]=1;
        helper(nrow,ncol,n,maze,temp,ans);
        temp[row][col]=0;
     }
  }

}
vector<vector<int> > ratInAMaze(vector<vector<int> > &maze, int n){
  // Write your code here.
  vector<vector<int>> ans;
  vector<vector<int>> temp(n,vector<int>(n,0));
  helper(0,0,n,maze,temp,ans);
  return ans;
}
```

- Time Complexity : O(4^ m*n)
- Space Complexity : O(m*n) depth and res vector.