```c
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>

#define POINT '.'
#define White 0
#define Gray  1
#define Black 2
#define MAX 2147483647

typedef struct {
  char key;
  int color;
} Vertex;

bool safe_vertex(Vertex **grid, int l, int c) {
  return (grid[l][c].color == White && grid[l][c].key == POINT) ? true
: false;
}

int solves = 0;

int visit(Vertex **grid, int n, int l, int c, bool brute_force) {
    if(l==n-1 && c==n-1)
        solves+=1;

    if(brute_force && solves > 0)
        return 1;

    if(brute_force)
        grid[l][c].color = Gray;

    if(c<n-1 &&safe_vertex(grid, l, c+1))
        visit( grid, n, l, c+1, brute_force);

    if(l<n-1 && safe_vertex(grid, l+1, c))
        visit(grid, n, l+1, c, brute_force);

    if(brute_force && c>0 && safe_vertex(grid, l, c-1))
        visit(grid, n, l, c-1, brute_force);

    if(brute_force && l>0 && safe_vertex(grid, l-1, c))
        visit(grid, n, l-1, c, brute_force);
}

int main() {
  int n, i, j;
  char c;

  //Dimensão da malha
  scanf("%d", &n);
  getchar();

  Vertex **grid = (Vertex**)malloc(n*sizeof(Vertex*));

  for(i=0; i<n; i++) {
        grid[i] = (Vertex*)malloc(n*sizeof(Vertex));
    }

    for(i=0;i<n;i++) {
        for(j=0; j<n;j++){
            scanf("%c", &c);
            grid[i][j].key = c;
            grid[i][j].color = White;
        }
        getchar();
    }

    if(n<=13) {
        visit(grid, n, 0, 0, false);
        if(solves>0)
          printf("%d\n", solves%MAX);
        else {
          visit(grid, n, 0, 0, true);

          if(solves==0)
              printf("INCONCEIVABLE\n");
          else
              printf("THE GAME IS A LIE\n");
        }
    } else {
          visit(grid, n, 0, 0, true);

          if(solves==0)
              printf("INCONCEIVABLE\n");
          else
              printf("THE GAME IS A LIE\n");
    }

    return 0;
}
```