实验二：深度优先和广度优先搜索

一．实验目的

掌握深度优先遍历和广度优先遍历的思想、实现栈以及队列的基本操作。

二．实验内容

(1) 实验 1

输入一个任意大小的迷宫，用**深度优先遍历(用栈的方法实现)**的方法求出**从起点开始能走到的格数(不包含起点)，并输出所有能到达的点**。

在输入中，第一行输入两个用空格隔开的整数 n 和 m(1<=n, m<=100)，代表迷宫的行数和列数。接下来用“#”代表墙壁，“0”代表空地，“?”表示起点。迷宫只会有一个起点。假设每一步可以向上下左右四个方向移动 1 格。

在输出中，用“$”覆盖所有能到达的点。

样例如下表所示，在输入的迷宫中，从起点能到达的只有第二行的 4 个点，因此输出 4。

输入	5 5 ##### ?0000 ##### 000## ###00
输出	4 ##### ?$$$$ ##### 000## ###00

(2) 实验 2

输入一个任意大小的迷宫，用广度优先遍历(用队列的方法实现)的方法求出走出迷宫的最短路径长度(步数)，并将路径输出。

在输入中，第一行输入两个用空格隔开的整数 n 和 m(1<=n, m<=100)，代表迷宫的行数和列数。接下来用“#”代表墙壁，“0”代表空地，“?”表示起点。迷宫只会有一个起点。假设每一步可以向上下左右四个方向移动 1 格。

在输出中，用“*”覆盖最短路径的点。如果有多条最短路，输出任意一条即可。当没有可行路径时，输出“impossible”。

例：

输入	7 10 ########## #?0#000#0# #000000#0# #00#00000# ########0# ###000000# #####0####	输出 (不唯一)	16 ########## #?*#000#0# #0*****#0# #00#00***# ########*# ###00****# #####*####

三．实验要求

1） 基于参考代码 main.c 操作；
2） 从文件中读取迷宫数据；
3） 请用手写的栈和队列实现，即你的代码中要包含栈和队列的实现。若未实现可能会扣分；
4） 要求在实验课上完成（1）或（2），在课下完成全部实验，并撰写实验报告

四．参考代码

```
#include <stdio.h>
#include <malloc.h>
#include <string.h>
#define maxn 1002
/*
maze 数组用于存储迷宫
vis 数组用于在遍历的时候标记已经访问过的点
dir 数组用于表示上下左右方向
n 和 m 分别表示迷宫的行数和列数
*/
char maze[maxn][maxn];
int vis[maxn][maxn];
int dir[4][2] = {{0,1},{1,0},{0,-1},{-1,0}};
int n,m;

int LegalPosition(int x, int y){
/* 判断(x,y)位置是否合法*/

}

typedef struct snode{
/* 栈中存储的节点 */
    int x; // 行坐标
    int y; // 列坐标
    int flag; // 当前方向
    struct snode *prev; // 栈中上一个节点的指针
/* 可自由添加需要用的变量 */

}StackNode;

typedef struct {
/* 栈 */
    StackNode *top; // 栈顶指针
```

```
    int size; // 栈中节点个数
/* 可自由添加需要用的变量 */

}Stack;

Stack *InitStack(){
/* 初始化栈 */
    Stack* sta = (Stack *)malloc(sizeof(Stack));
    sta->top = NULL;
    sta->size = 0;
}
int isEmpty(Stack *sta){
/* 判断栈是否为空 */

}
int Push(Stack *sta, StackNode now){
/* 將一个节点压入栈中 */

}
StackNode *Pop(Stack *sta){
/* 將一个节点弹出栈 */

}

void DFS(char maze[maxn][maxn], int n, int m){
/*
  深度优先遍历的过程
  请在本过程中求出从"?"能到达的点数
  并将所有能到达的点用"$"覆盖
*/
    Stack *q = InitStack();
    memset(vis, 0, sizeof(vis));

}


typedef struct qnode{
/* 队列中存储的节点 */
    int x; // 行坐标
    int y; // 列坐标
    int step; // 当前点与起点的距离
    struct qnode *succ; //队列中下一个节点的指针
/* 可自由添加需要用的变量 */
```

```
}QueueNode;

typedef struct {
/* 队列 */
    QueueNode *front; //队首指针
    QueueNode *back; //队尾指针
    int size; // 队列中节点个数
/* 可自由添加需要用的变量 */

}Queue;

Queue *InitQueue(){
/* 初始化队列 */
    Queue* que = (Queue *)malloc(sizeof(Queue));
    que->front = NULL;
    que->back = NULL;
    que->size = 0;
}
int isEmpty(Queue *que){
/* 判断队列是否为空 */

}
int EnQueue(Queue *que, QueueNode now){
/* 入队 */

}
QueueNode *DeQueue(Queue *que){
/* 出队 */

}


void BFS(char maze[maxn][maxn], int n, int m){
/*
  广度优先遍历的过程
  请在本过程中求出从"?"出发走出迷宫的最短路长度
  并将最短路径用"*"覆盖
*/
    Queue *q = InitQueue();
    memset(vis, 0, sizeof(vis));


}
```

```
int main()
{
/* 从文件 maze.in 中读入 */
    // freopen("maze.in","r",stdin);
    int o = 0;
    while (scanf("%d%d",&n,&m)!=EOF){
    // 可能有多组输入数据
        o += 1;
        for (int i=1;i<=n;i++)
            for (int j=1;j<=m;j++)
                scanf(" %c", &maze[i][j]);
        printf("--------Case #%d--------\n", o);
        DFS(maze, n, m);
        BFS(maze, n, m);
    }
    // fclose(stdin);
    return 0;
}
```

五．范例说明

第一个样例中，从?点起，只能向右走 4 个格子。
第二个样例中，最少要走 16 步才能走出迷宫，最短路线已经用*标出。

实验报告提交说明：

1．电子版实验报告提交的截止时间为：2019 年 4 月 27 日晚上 10 点之前；
2．电子版实验报告提交到该邮箱：hitsz_ds_2019@163.com；
3．请把电子版实验报告及源代码打包成一个压缩包，命名格式如下：
   a) 实验报告：“学号_姓名_实验 2”
   b) 压缩包：“学号_姓名_实验 2”
   c) 邮件标题：“学号_姓名_实验 2”
4．纸质版实验报告请在 4 月 28 日晚上 9 点之前提交到 G701。