## 实验 老鼠走迷宫

## **1、需求分析**

利用二维数组创建一个随机迷宫或者自定义迷宫，再在迷宫中找到一条从起点到终点的路径，并用图形界面显示。

**2、概要设计**



**3、详细设计**

输入迷宫行数列数

创建迷宫外墙

If（随机迷宫）

{

调用创建随机迷宫函数，创建随机迷宫

}

else（自定义迷宫）

{

调用自定义迷宫函数，创建自定义迷宫

}

调用递归访问迷宫函数，寻找一条从入口到出口的路径

显示迷宫的图形界面

**4、调试分析**

该程序对迷宫行列数操作时一定要注意分清楚到底是对行操作还是对列操作，此处易混淆；递归访问函数较复杂；随机迷宫如果需要创建成功通过的迷宫需多次尝试。

**5、用户使用说明**

用户首先根据提示输入迷宫行数列数，再选择创建随机迷宫还是自定义迷宫（1为随机迷宫，2为自定义迷宫）。

如果创建随机迷宫输入1，接下来程序会自动创建随机迷宫，并对迷宫进行寻找出口操作，有出口则显示含有出口路径的迷宫图，没有出口则显示：没有找到出口！

如果创建自定义迷宫，则根据提示输入代表墙壁的2或者代表通道的0组成迷宫，

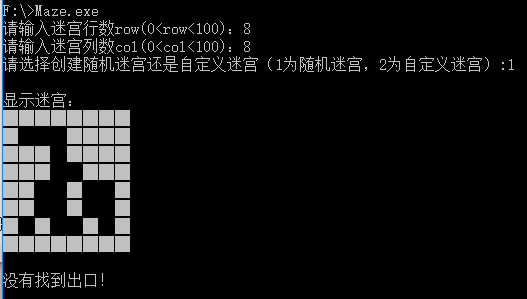
有出口则显示含有出口路径的迷宫图，没有出口则显示：没有找到出口！

1. **运行结果**

6.1有出口的自定义迷宫



6.2没有出口的随机迷宫

****

6.3有出口的随机迷宫

****

**7、源程序**  
#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int createMaze(); //创建随机迷宫

int createFreeMaze(); //创建自定义迷宫

void createWall(); //创建迷宫外墙

int visit(int row2, int col2) ;

int startI = 1, startJ = 1; // 入口

int success = 0;

//迷宫数组

int maze[100][100];

int row = 0;

int col = 0;

//迷宫矩阵，2代表墙壁，0代表通道

//创建迷宫外墙

void createWall()

{

//创建迷宫外墙,第一行、第一列、最后一行、最后一列均为墙壁

for(int i = 0; i < col; i++)//第一行

maze[0][i] = 2;

for(int i = 1; i < row; i++)//第一列

maze[i][0] = 2;

for(int i = 1; i < col; i++)//最后一行

maze[row-1][i] = 2;

for(int i = 1; i < row-1; i++)//最后一列

maze[i][col-1] = 2;

}

//创建随机迷宫

int createMaze()

{

srand(time(0));

for(int i = 1; i < row-1; i++)

{

for(int j = 1; j < col-1; j++)

{

if((rand()%100+1) % 2 == 0)

maze[i][j] = 0;

else

maze[i][j] = 2;

}

}

maze[1][1] = 0;

maze[row-2][col-2] = 0;

}

//创建自定义迷宫

int createFreeMaze()

{

for(int i = 1; i < row-1; i++)

{

//第一行，第一格为入口

if(i == 1)

{

printf("迷宫第%d行，共%d格： ",i,col-3);

for(int j = 2; j < col-1; j++)

scanf("%d",&maze[i][j]);

}

//最后一行,最后一格为出口

else if(i == row-2)

{

printf("迷宫第%d行，共%d格：",i,col-3);

for(int j = 1; j < col-2; j++)

scanf("%d",&maze[i][j]);

}

else

{

printf("迷宫第%d行，共%d格：",i,col-2);

for(int j = 1; j < col-1; j++)

scanf("%d",&maze[i][j]);

}

}

maze[1][1] = 0; //入口为通道

maze[row-2][col-2] = 0; //出口为通道

}

int visit(int row2, int col2)

{

int endI = row-2, endJ = col-2; // 出口

//该点走过，标记为1

maze[row2][col2] = 1;

//走到终点，成功

if(row2 == endI && col2 == endJ)

success = 1;

//向四个方向递归调用函数visit()

if(success != 1 && maze[row2][col2+1] == 0)

visit(row2, col2+1);

if(success != 1 && maze[row2+1][col2] == 0)

visit(row2+1, col2);

if(success != 1 && maze[row2][col2-1] == 0)

visit(row2, col2-1);

if(success != 1 && maze[row2-1][col2] == 0)

visit(row2-1, col2);

//该点走过，但没成功，则该点重新置为0

if(success != 1)

maze[row2][col2] = 0;

return success;

}

int main(void)

{

int i, j;

printf("请输入迷宫行数row(0<row<100)：");

scanf("%d",&row);

printf("请输入迷宫列数col(0<col<100)：");

scanf("%d",&col);

createWall();//创建迷宫外墙

int choice;

printf("请选择创建随机迷宫还是自定义迷宫（1为随机迷宫，2为自定义迷宫）:");

scanf("%d",&choice);

if(choice == 1)

{

createMaze(); //创建迷宫

}

else if(choice == 2)

{

printf("\n请输入自定义迷宫的墙壁和通道，2代表墙壁，0代表通道\n");

createFreeMaze();

}

printf("\n显示迷宫：\n");

for(i = 0; i < row; i++)

{

for(j = 0; j < col; j++)

{

if(maze[i][j] == 2)

printf("█");

else

printf(" ");

}

printf("\n");

}

if(visit(startI, startJ) == 0)

{

printf("\n没有找到出口！\n");

}

else

{

printf("\n显示路径：\n");

for(i = 0; i < row; i++)

{

for(j = 0; j < col; j++)

{

if(maze[i][j] == 2)

printf("█");

else if(maze[i][j] == 1)

printf("◇");

else

printf(" ");

}

printf("\n");

}

}

system("pause");

return 0;

}