项目说明文档

数据结构课程设计

——勇闯迷宫游戏

作 者 姓 名： 陈君

学 号： 2250420

指 导 教 师： 张颖

学院、 专业： 软件学院 软件工程

同济大学

Tongji Universit

# 1 分析

## 1.1 背景分析

在一个遥远的大陆，有一个古老而神秘的迷宫。这个迷宫据说隐藏着一个古代文明的宝藏。但是，数百年来，无数勇者尝试进入迷宫寻找宝藏，但很少有人能够走出迷宫。那里充满了各种陷阱和怪物，需要勇者不仅拥有出色的战斗技巧，还需要智慧和观察力，才能够成功地找到宝藏并离开迷宫。现在要你写一个程序帮助他找到出口。

## 1.2 功能分析

# 迷宫问题的求解过程可以采用回溯法即在一定的约束条件下试探地搜索前进，若前进中受阻，则及时回头纠正错误另择通路继续搜索的方法。从入口出发，按某一方向向前探索，若能走通，即某处可达，则到达新点，否则探索下一个方向；若所有的方向均没有通路，则沿原路返回前一点，换下一个方向再继续试探，直到所有可能的道路都探索到，或找到一条通路，或无路可走又返回入口点。在求解过程中，为了保证在达到某一个点后不能向前继续行走时，能正确返回前一个以便从下一个方向向前试探，则需要在试探过程中保存所能够达到的每个点的下标以及该点前进的方向，当找到出口时试探过程就结束了。（地图写死）

# 2 设计

## 2.1 数据结构设计

如上功能分析所述，该游戏需要用到一个二维的迷宫，所以数据结构应该使用二维数组存储迷宫比较好，可以直接获取迷宫各个点的坐标。

## 2.2 类结构设计

本系统采用设计maze类和结构体point，maze用于存储迷宫的信息已经有关反映迷宫的方法，结构体point存储当前的位置。

## 2.3 成员与操作设计

**迷宫类（listnode）**

class Maze

{

public:

void show();//展示迷宫

bool isOutter(Point point) { return map[point.x][point.y] == outer; }//判断是否在出口

bool iswall(Point point) {return map[point.x][point.y] == wall;}//判断该位置是否是墙

bool ismark(Point point) { return map[point.x][point.y] == mark; }//判断这个位子是不是被标记了

void markroad(Point point) { map[point.x][point.y] = mark; }//将这个位置做标记

void rmvmark(Point point) { map[point.x][point.y] = road; }//将这个位置标记删除

private:

friend Move;

char map[7][7] = { {'#','s','#','#','#','#','#'}//地图

,{'#','0','#','0','0','0','#'}

,{'#','0','#','0','#','#','#'}

,{'#','0','0','0','#','0','#'}

,{'#','0','#','0','0','0','#'}

,{'#','0','#','0','#','0','#'}

,{'#','#','#','#','#','o','#'}

};

};

结构体Point

struct Point

{

int x ;

int y ;

};

## 2.4 系统设计

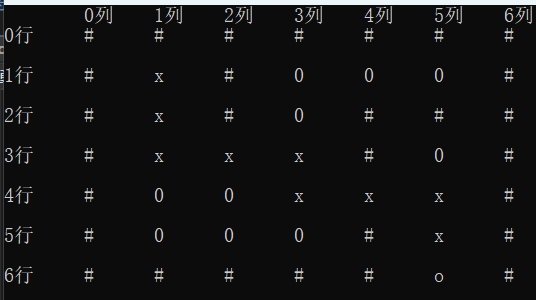
主程序先声明一个maze对象创建迷宫和一个结构体point用于存储点的位置，然后递归调用findway函数找出可以走出迷宫的一条道路，然后调用show方法显示含有正确路径的迷宫，再调用showtrack，显示这条正确的路径。

# 3 实现

## 3.1 void show();//展示迷宫

先输出迷宫的列： 0列 1列 2列 3列 4列 5列 6列。然后使用两个for循环遍历迷宫的每个格子然后打印。

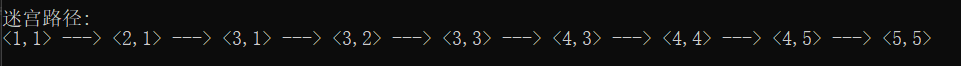
### 3.2.3 构造函数功能截屏示例



## 3.2 void Maze::showtrack()功能的实现

具体实现：使用两个for循环遍历每个格子上的点，如果这个格子上的字符是‘x’代表是标记的路，就打印这个点的坐标。

### 3.2.3 count功能截屏示例



## 3.3 bool isOutter(Point point) { return map[point.x][point.y] == outer; }//判断是否在出口

从头结点开始，打印结点中存储的位置，然后将p结点指向下一个位置，知道p结点的下一个结点为first为止。

### 3.3.2 putout功能截图示例

## 3.4 bool isOutter(Point point) { return map[point.x][point.y] == outer; }//判断是否在出口

返回（map[point.x][point.y] == outer;）的值

3.5 bool iswall(Point point) {return map[point.x][point.y] == wall;}//判断该位置是否是墙

返回（map[point.x][point.y] == wall;）的值

截图展示：



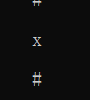
## 3.6 bool ismark(Point point) { return map[point.x][point.y] == mark; }//判断这个位子是不是被标记了

返回（map[point.x][point.y] == mark;）的值

## 3.7 void markroad(Point point) { map[point.x][point.y] = mark; }//将这个位置做标记

## map[point.x][point.y] = mark;

截图展示：



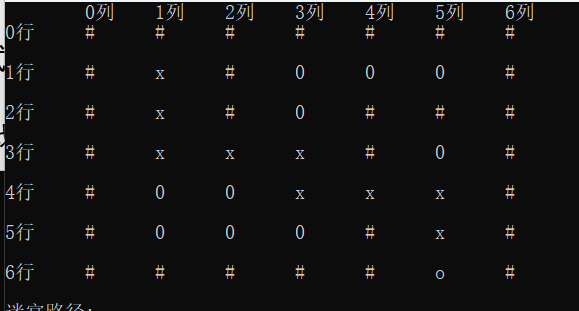
## 3.8void rmvmark(Point point) { map[point.x][point.y] = road; }//将这个位置标记删除

map[point.x][point.y] = road;

## 3.6 总体系统的实现

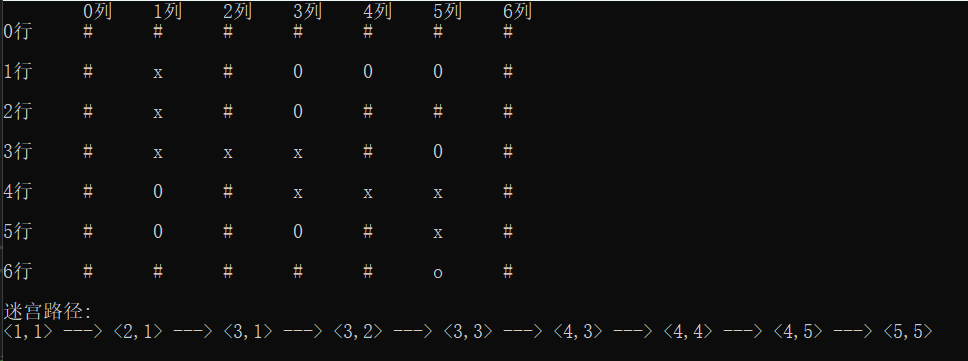
主程序先声明一个maze对象创建迷宫和一个结构体point用于存储点的位置，然后递归调用findway函数找出可以走出迷宫的一条道路，然后调用show方法显示含有正确路径的迷宫，再调用showtrack，显示这条正确的路径。

### }3.6.1 总体系统截屏示例

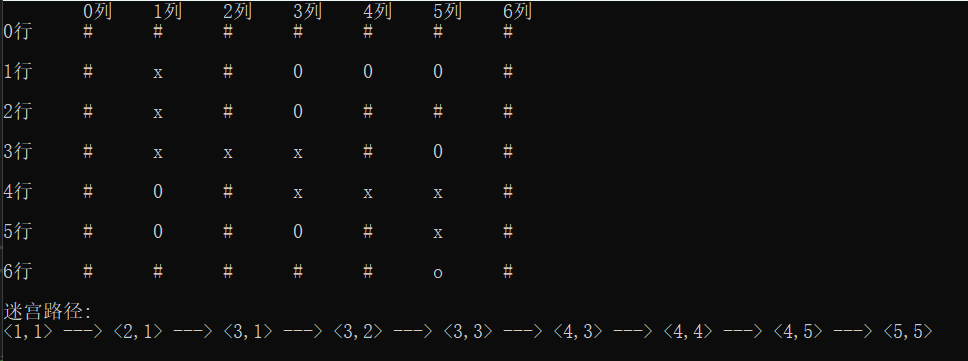


# 4 测试

## 4.1 游戏总体功能测试功能测试



### 4.1.1 截屏展示



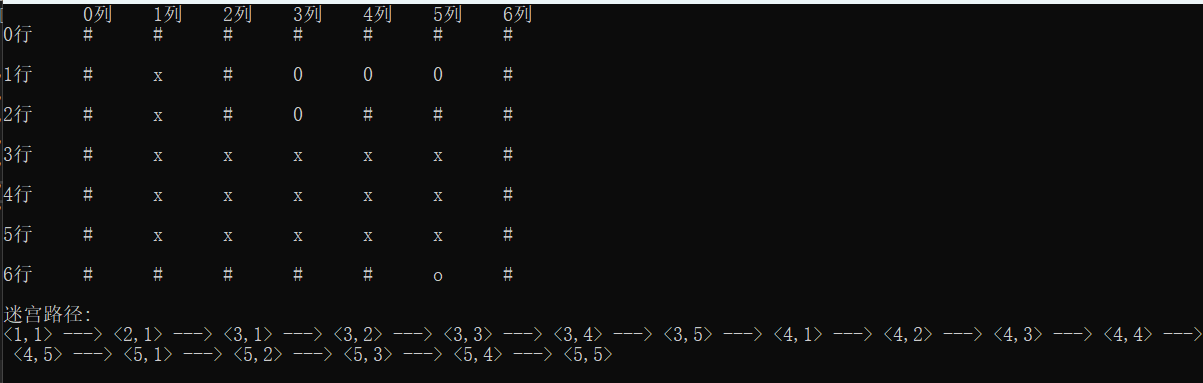
4.2 输入无法走出的迷宫

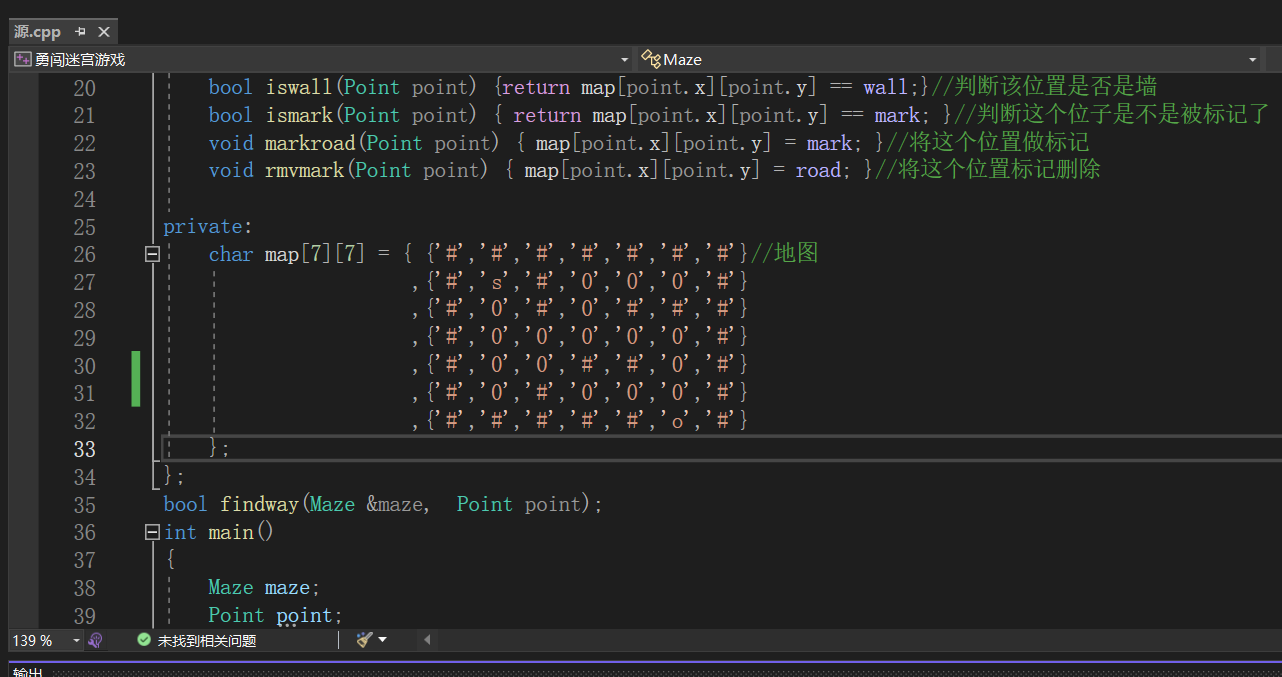


4.2.1截屏展示



4.3 输入有多个正确路径的迷宫

4.3.1 截屏展示

第二个测试用例

