勇闯迷宫游戏

问题描述

迷宫只有两个门，一个门叫入口，另一个门叫出口。一个骑士骑马从入口进入迷宫，迷宫设置很多障碍，骑士需要在迷宫中寻找通路以到达出口。

项目要求

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。

算法设计

回溯法求解迷宫路径，用顺序结构实现栈完成回溯。

功能实现及代码分析

1、结构体设计

栈中结点保存当前方块的行、列以及可走方位。

```
struct Try //定义一个栈，保存路径
{
    int curX;            //当前方块的行号
    int curY;            //当前方块的列号
    int dir;             //dir是下一可走方位的方位号
} path[MaxSize];         //定义栈
```

2、创建迷宫

进入游戏，引导用户输入自定义迷宫的行列数，出发、终点位置，并输入地图详细内容，若行、列任一为0则使用默认地图。

```
int main() {
    int row, col;                                   //行数和列数
    cout << "请设置迷宫的行数和列数（任一为0则使用默认地图）：";
    cin >> row >> col;
    if (row != 0  && col != 0) {
        cout << "请设置" << row << '*' << col << "迷宫(1为墙壁，0为通道）：\n";
        for (int i = 0; i < row; i++) {             //输入迷宫数据
            for (int j = 0; j < col; j++) {
                cin >> maze[i][j];
            }
        }
        display(row, col);  //输出迷宫地图
    } else {
        display(7, 7);
    }
    //...
}
```

3、寻找出口

1. 在第n步操作中，按顺序调整方向，寻找下一个可走方块，压入栈中，标记为已走过防止重复，若路径无法达到终点，逐个弹出栈顶方块元素，直到出现可走路径，然后继续将下一方块压入栈的操作。
2. 找到通往终点的路径后，按顺序从栈底到栈顶逐个输出方块行列坐标，得到迷宫路径。

```
int main() {
    //...
```

```
    int curX, curY, dir, find, k;
    top++;                                          //初始方块进栈
    path[top].curX = xBegin;
    path[top].curY = yBegin;
    path[top].dir = -1;
    maze[xBegin][yBegin] = -1;
    while (top > -1)  {                             //栈不为空时循环
        curX = path[top].curX;
        curY = path[top].curY;
        dir = path[top].dir;
        if (curX == xExit && curY == yExit) {       //找到了出口，输出路径
            cout << "迷宫路径如下：\n";
            for (k = 0; k <= top; k++) {
                cout << "(" << path[k].curX << "," << path[k].curY << ")";
                if (k != top) cout << "-->";
            }
            cout << endl;
            return 0;
        }
        find = 0;
        while (dir < 4 && find == 0) {              //找下一个可走的点
            dir++;
            switch (dir) {
                case 0:  //向上
                    curX = path[top].curX - 1;
                    curY = path[top].curY;
                    break;
                case 1: //向右
                    curX = path[top].curX;
                    curY = path[top].curY + 1;
                    break;
                case 2:  //向下
                    curX = path[top].curX + 1;
                    curY = path[top].curY;
                    break;
                case 3:  //向左
                    curX = path[top].curX;
                    curY = path[top].curY - 1;
                    break;
            }
            if (maze[curX][curY] == 0) find = 1;     //找到通路
        }
        if (find == 1) {                            //找到了下一个可走方块
            path[top].dir = dir;                    //修改原栈顶元素的d值
            top++;                                  //下一个可走方块进栈
            path[top].curX = curX;
            path[top].curY = curY;
            path[top].dir = -1;
            maze[curX][curY] = -1;                  //避免重复走到这个方块
        } else {                                    //没有路可走，则退栈
            maze[path[top].curX][path[top].curY] = 0;//让该位置变成其它路径可走方块
            top--;
        }
    }
    cout << "没有可走路径！\n";
    system("pause");
    return 0;
}
```

用例演示

默认地图

```
请设置迷宫的行数和列数（任一为0则使用默认地图）：0 1
迷宫地图：
        第0列   第1列    第2列    第3列    第4列    第5列    第6列
第0行    #       #       #       #       #       #       #
第1行    #       x       #       x       x       x       #
第2行    #       x       #       x       #       #       #
第3行    #       x       x       x       #       x       #
第4行    #       x       #       x       x       x       #
第5行    #       x       #       x       #       x       #
第6行    #       #       #       #       #       #       #
输入起点(分别输入横纵坐标)：1 1
输入出口(分别输入横纵坐标)：5 5
```

```
迷宫路径如下:
(1,1)-->(2,1)-->(3,1)-->(3,2)-->(3,3)-->(4,3)-->(4,4)-->(4,5)-->(5,5)

Process finished with exit code 0
```

自定义地图

```
请设置迷宫的行数和列数（任一为0则使用默认地图）： 5 5
请设置5*5迷宫(1为墙壁，0为通道）：
0 1 1 1 1
0 0 0 0 0
1 0 1 1 0
1 0 1 1 1
1 0 0 0 1
迷宫地图:
        第0列    第1列    第2列    第3列    第4列
第0行     x       #       #       #       #
第1行     x       x       x       x       x
第2行     #       x       #       #       x
第3行     #       x       #       #       #
第4行     #       x       x       x       #
输入起点(分别输入横纵坐标）： 0 0
输入出口(分别输入横纵坐标）： 4 3
迷宫路径如下:
(0,0)-->(1,0)-->(1,1)-->(2,1)-->(3,1)-->(4,1)-->(4,2)-->(4,3)

Process finished with exit code 0
```