实验题目：

广度优先搜索解决迷宫问题

实验代码：

#include<iostream>

#include<ctime>

#include<cstdlib>

#include<queue>

#include<cstdio>

using namespace std;

//生成迷宫

int isFound = 0;

int maze[10][10] = {0,0,0,1,1,0,1,0,1,1,

1,0,0,1,0,0,1,0,0,0,

1,1,0,1,0,1,0,1,0,1,

0,1,0,1,0,1,1,0,1,0,

0,0,1,1,0,0,0,1,0,0,

1,1,0,1,0,0,1,0,0,1,

1,1,1,0,1,1,0,0,1,1,

1,1,0,0,1,1,0,0,0,0,

0,1,0,0,1,1,0,0,0,1,

0,1,1,1,1,1,0,0,0,0};

void printMaze()

{

maze[0][0] = 0;//入口

maze[9][9] = 0;//出口

//输出迷宫

for (int i = 0; i < 10; i++)

{

for (int j = 0; j < 10; j++)

{

cout << maze[i][j];

if (j != 9)

{

cout << " ";

}

else

{

cout << endl;

}

}

}

}

//生成方向

int direct[8][2] = {{0,1},{0,-1},{1,1},{1,-1},{-1,1},{-1,-1},{1,0},{-1,0}};

//判断是否越界

bool isLeap(int x, int y)

{

return x >= 0 && x < 10&&y >= 0 && y < 10;

}

//任意位置的结构体

struct point {

int x;

int y;

};

//声明用于存储路径的结构体

struct dir

{

int x;

int y;

int d;

};

//声明用于存储路径的队列

queue<dir> directoryQueue;

//迷宫循迹

dir path[10][10];//记录迷宫的路径

int output[100][3];

void mazeTravel(point start, point end, int maze[10][10], int direct[8][2])

{

dir element;

int i;

int j;

int d;

int a;

int b;

element.x = start.x;

element.y = start.y;

element.d = -1;

maze[start.x][start.y] = 2;

directoryQueue.push(element);

while (!directoryQueue.empty())

{

element = directoryQueue.front();

dir m = element;

directoryQueue.pop();

i = element.x;

j = element.y;

d = element.d + 1;

while (d < 8)

{

a = i + direct[d][0];

b = j + direct[d][1];

if (a == end.x&&b == end.y&&maze[a][b] == 0)

{

//储存前一个点的信息至path

dir temp = m;

temp.d = d;

path[a][b] = temp;

isFound = true;

return;

}

if (isLeap(a,b)&&maze[a][b]==0)

{

//储存前一个点的信息至path

dir temp = m;

temp.d = d;

path[a][b] = temp;

maze[a][b] = 2;

element.x = a;

element.y = b;

element.d = -1;

directoryQueue.push(element);

}

d++;

}

}

}

void printPath(point start, point end)

{

if (!isFound)

printf("The path is not found");

else

{

int step = 0;

dir q;

q.x = end.x;

q.y = end.y;

q.d = 123;

while (q.x != start.x || q.y != start.y)

{

output[step][0] = q.x;

output[step][1] = q.y;

output[step][2] = q.d;

int x = q.x;

int y = q.y;

q.x = path[q.x][q.y].x;

q.y = path[x][q.y].y;

q.d = path[x][y].d;

step++;

}

output[step][0] = q.x;

output[step][1] = q.y;

output[step][2] = q.d;

printf("The path is as follows: \n");

for (int i = step; i >= 0; i--)

{

printf("(%d,%d)", output[i][0], output[i][1]);

if (i != 0)

printf("->");

}

printf("\n");

}

}

int main()

{

printMaze();

point a, b;

a.x = 0;

a.y = 0;

b.x = 9;

b.y = 9;

mazeTravel(a, b, maze, direct);

printPath(a, b);

return 0;

}

运行结果：

