2020 年 8 月 3 日

1.P1605 迷宫

算法思路：

dfs+回溯。从起始坐标开始dfs，到达终点则结果+1，返回。dfs函数：当前节点置为访问过，然后按照上下左右查看是否符合条件，符合条件的继续dfs，最后要将前面设置为访问过的重置为初始状态，以便回溯。

代码：

```
#include <iostream>

using namespace std;

#define N 10
#define M 10

int mazes[N][M] = {0};
int n,m,t,sx,sy,fx,fy,res = 0;

void dfs(int row,int col){
    if(row == fx && col == fy){
        res++;
        return;
    }
    mazes[row][col] = 1;    //访问过
    if( (row-1 >= 1) && mazes[row-1][col] == 0) dfs(row-1,col); //上
    if( (row+1 <= n) && mazes[row+1][col] == 0) dfs(row+1,col); //下
    if( (col-1 >= 1) && mazes[row][col-1] == 0) dfs(row,col-1); //左
    if( (col+1 <= m) && mazes[row][col+1] == 0) dfs(row,col+1); //右
    mazes[row][col] = 0;    //重置
}

int main(){

    cin >> n >> m >> t;
    cin >> sx >> sy >> fx >> fy;

    for(int i = 0; i < t; i++){
        int t1,t2;
        cin >> t1 >> t2;
        mazes[t1][t2] = 1;  //1是障碍
    }

    dfs(sx,sy);
```

```
    cout << res;
    return 0;
}
```

Accepted截图：

2.P1238 走迷宫

算法思路：

和上一道题类似，只是多了一步要记录路径和四个方向的访问顺序题目有要求。在这里用path记录当前的路径，则在每一次dfs前后，加入节点和删去节点，res记录总的路径。

代码：

```
#include <iostream>
#include <vector>
using namespace std;

#define N 20
#define M 20

int mazes[N][M];
int n,m,sx,sy,fx,fy;

void dfs(int row,int col,vector<int> path,vector< vector<int> > &res){
    if(row == fx && col == fy){
        res.push_back(path);
        return;
    }

    mazes[row][col] = 0;   //访问过
    if( col-1 >= 1 && mazes[row][col-1] == 1){      //左
        path.push_back(row);
        path.push_back(col-1);
        dfs(row,col-1,path,res);
```

```
            path.pop_back();
            path.pop_back();
        }
        if( row-1 >= 1 && mazes[row-1][col] == 1){      //上
            path.push_back(row-1);
            path.push_back(col);
            dfs(row-1,col,path,res);
            path.pop_back();
            path.pop_back();
        }
        if( col+1 <= m && mazes[row][col+1] == 1){      //右
            path.push_back(row);
            path.push_back(col+1);
            dfs(row,col+1,path,res);
            path.pop_back();
            path.pop_back();
        }
        if( row+1 <= n && mazes[row+1][col] == 1){      //下
            path.push_back(row+1);
            path.push_back(col);
            dfs(row+1,col,path,res);
            path.pop_back();
            path.pop_back();
        }
        mazes[row][col] = 1;   //重置
        return;
    }

    int main(){
        cin >> n >> m;
        for(int i = 1; i <= n; i++){
            for(int j = 1; j <= m; j++){
                int temp;
                cin >> temp;
                mazes[i][j] = temp;
            }
        }
        cin >> sx >> sy >> fx >> fy;

        vector< vector<int> > res;
        vector<int> path;
        path.push_back(sx);
        path.push_back(sy);
        dfs(sx,sy,path,res);

        for(int i = 0; i < res.size(); i++){
            int j = 0;
            cout << "(" << res[i][j] << "," << res[i][j+1] << ")";
            for( j=j+2; j < res[i].size(); j=j+2){
                cout << "->(" << res[i][j] << "," << res[i][j+1] << ")";
            }
            cout << endl;
        }
        if( res.size() == 0 ) cout << -1;
```

```
    return 0;
}
```

Accepted截图：

备注：

1. vector好像不能res[i][++j]，正确的写法是res[i][j+1]
2. 一开始忘了没有路径要输出-1，所以只得了80分~