1. 问题描述

2. 思路及实现代码

1. 问题描述

<http://acm.hrbeu.edu.cn/index.php?act=problem&id=1004&cid=18>

某石油公司发现了一个油田。该油田由n\*m个单元组成的矩形，有些单元里有石油，有些则没有。单元油田可以通过上，下，左或右连通。在一个单元油田里架设一台采油机，它可以把和该单元油田相连的单元油田的石油采完。该公司想知道最少需要架设几台采油机能把所有的石油采完？

示例输入：

2 2   
@\*

\*@

2 2  
@@  
@@

示例输出：

2

1

2. 思路及实现代码

广度优先搜索算法实现。

复制代码

// 广度优先搜索算法  
#include <iostream>  
#include <stack>  
#include <utility>  
  
using namespace std;  
  
int main()  
{  
    int oils = 0;  
    int lines = 2, rows = 2;  
    // 得到输入  
    std::cin >> rows;  
    std::cin >> lines;  
  
    // 定义地图  
    char map[rows][lines];  
    // 标志数组，并初始化，0表示没有标记  
    // 1表示原先不是油田，但是已经查找过，  
    // 2表示原先是油田，但已经查找过  
    int flag[rows][lines];  
    // 初始化  
    for(int i = 0; i < rows; ++i)  
        for(int j = 0; j < lines; ++j)  
        {  
            flag[i][j] = 0;  
            std::cin >> map[i][j];  
        }  
  
  
    stack<pair<int, int> > context;  
    int first, second;  
    for(int i = 0; i < rows; ++i)  
        for(int j = 0; j < lines; ++j)  
        {  
            // 如果已经标记，跳过  
            if (flag[i][j])  
                continue;  
            // 如果不是油田，标记，跳过  
            if (map[i][j] != '@')  
            {  
                flag[i][j] = 1;  
                continue;  
            }  
  
            // 没有标记过，并且是油田  
            // 标记  
            // flag[i][j] = 2;  
            oils++; // 增加数量  
            context.push(make\_pair(i, j));  
            // 栈不空  
            while(!context.empty())  
            {  
                // 弹栈，标记元素  
                pair<int, int> pos = context.top();  
                context.pop();  
                first = pos.first;  
                second = pos.second;  
  
                flag[first][second] = 2;  
  
                // 查找四个方向的油田，如果查找到  
                // 标记，并将该油田的位置插入到栈  
                // 中  
                if ( ( (first - 1) >= 0 ) &&  
                    (map[first - 1][second] == '@') &&  
                    (!flag[first - 1][second]))  
                    {  
                        context.push(make\_pair(first - 1, second));  
                    }  
                 if ( ( (second - 1) >= 0 ) &&  
                    (map[first][second - 1] == '@') &&  
                     (!flag[first][second - 1]))  
                    {  
                        context.push(make\_pair(first, second - 1));  
                    }  
                 if ( ( (first + 1) < rows ) &&  
                    (map[first + 1][second] == '@') &&  
                     (!flag[first + 1][second]) )  
                    {  
                        context.push(make\_pair(first + 1, second));  
                    }  
                 if ( ( (second + 1) < lines ) &&  
                    (map[first][second + 1] == '@') &&  
                     (!flag[first][second + 1]))  
                    {  
                        context.push(make\_pair(first, second + 1));  
                    }  
            }  
        }  
  
    cout << oils << endl;  
  
    return 0;

}

复制代码