链滴

# PAT 甲级刷题实录——1013

作者：aopstudio
原文链接：https://ld246.com/article/1580027336745
来源网站：链滴

# 原题链接

https://pintia.cn/problem-sets/994805342720868352/problems/994805500414115840

# 思路

题目大意是说一些城市之间有路相通，假设其中一个城市被敌方占领了，计算需要新修多少条路才能剩下的城市全部联通。首先这是一个典型的图论问题，我们可以用邻接矩阵去存城市之间的联通关系可以用深度遍历的思想去解决这个问题。思路大意如下：建立一个数组存储哪些城市已经被联通，1表已联通，0代表未联通；定义一个变量记录需要新修的路的数量，该变量初值为-1。对所有结点进行or循环遍历，遍历到一个结点时，如果该节点还未联通，则需要新修的路的数量加1，并对该结点进深度遍历以更新联通数组。变量初值为-1而不是0的原因是因为这个算法记录了和被攻占城市联通的条路，需要去除掉该路。

# 代码

```
#include <iostream>
#include <vector>
using namespace std;
void updateReached(int start);
vector<int> reached;
vector<int> cWay;
vector<vector<int> > ways;
int main()
{
    int N, M, K;
    cin >> N >> M >> K;
    cWay.assign(N + 1, 0);
    ways.assign(N + 1, cWay);
    vector<int> result;
    for (int i = 0; i < M; i++)
    {
        int c1, c2;
        scanf("%d %d",&c1,&c2);
        ways[c1][c2] = 1;
        ways[c2][c1] = 1;
    }
    for (int i = 0; i < K; i++)
    {
        int newWayNum = -1;
        reached.assign(N + 1, 0);
        int checked;
        scanf("%d",&checked);
        reached[checked] = 1;
        for (int j = 1; j < N+1; j++)
        {
            if (reached[j] == 0)
            {
                updateReached(j);
                newWayNum++;
            }
        }
```

```
        printf("%d\n",newWayNum);
    }
}
void updateReached(int start)    //更新start可达的城市到reach列表
{
    reached[start] = 1;
    for (int i = 1; i < reached.size(); i++)
    {
        if (ways[start][i] == 1 && reached[i] == 0)    //有路通且还没加入reached列表
        {
            updateReached(i);
        }
    }
    return;
}
```

# 坑

这道题很坑的一点是一开始最后一个测试点一直运行超时，我还以为是算法逻辑出问题了，一直在找里有问题，找了半天找不出来。后来上网一查发现不少人都有运行超时的问题，原因是因为用了C++格的输入输出方式cin和cout，这种输入输出方式的效率比scanf和printf低。解决方法就是改成scanf printf