软件包管理器

一. 考察内容：

图论 连通性

二. 题目分析：

[题目大意]

题目为一个初始无边有向图给出一些加边操作，询问添加到多少条边之后图出现环。

[写题思路]

考虑离线处理操作，题目给出了一些强制在线的限制，后续答案由之前答案得出，但是由于当图出现环之前，对于每个操作的答案是恒定为1的，所以我们默认每个操作的上一个操作的答案都是1。先将所有操作读入，然后二分答案，对于选定的答案，进行建边，如果出现环，则缩小答案，直到二分出正确答案，然后按照要求，输出1和0即可。

三. 代码实现：

```
#define _CRT_SECURE_NO_DEPRECATE
/************************
*创建时间：2018 08 15
*文件类型：源代码文件
*题目来源：牛客
*当前状态：已通过
*备忘录：Tarjan 强连通分量
*作者：HtBest
************************/
#include <stdio.h>
#include <iostream>
#include <string>
#include <stdlib.h>
#include <string.h>
#include <algorithm>
#include <queue>
using namespace std;
#define MAXN 100001
int n,m,head[MAXN],_edge,dfsx[MAXN],_dfsx,low[MAXN],scc[MAXN],_scc;
struct EDGE
{
    int a,b,next;
    EDGE(int a=0,int b=0,int next=0):a(a),b(b),next(next){}
}edge[2*MAXN],a[2*MAXN];
deque <int> sta;
/* Variable explain:
n:节点数
m:边数
a:将操作离线
head:每个节点的第一条出边
edge:邻接表存边
_edge:邻接表元素个数
dfsx:dfs序
_dfsx:dfs序标记
low:每个节点子树可以访问到的最小dfs序
scc:保存每个强连通分量
_scc:强连通分量标记
```

```
sta:保存强连通分量的栈
*/
void adde(int a,int b)
{
    edge[++_edge]=EDGE(a,b,head[a]);
    head[a]=_edge;
}
void clear()
{
    _edge=_dfsx=_scc=0;
    sta.clear();
    for(int i=1;i<=n;++i)head[i]=dfsx[i]=low[i]=scc[i]=0;
}
void read()
{
    // freopen(".in","r",stdin);
    // freopen(".out","w",stdout);
    scanf("%d%d",&n,&m);
    if(m)scanf("%d%d",&a[1].a,&a[1].b),a[1].a=(a[1].a-2+n)%n+1,a[1].b=(a[1].b-2+n)%n+1;
    for(int i=2;i<=m;++i)scanf("%d%d",&a[i].a,&a[i].b),a[i].a=(a[i].a-3+n)
%n+1,a[i].b=(a[i].b-3+n)%n+1;
    return;
}
bool dfs(int a)
{
    dfsx[a]=low[a]=++_dfsx;
    sta.push_back(a);
    for(int i=head[a];i;i=edge[i].next)
    {
        int b=edge[i].b;
        if(!dfsx[b])
        {
            if(dfs(b))return true;
            low[a]=min(low[a],low[b]);
        }
        else if(!scc[b])
        {
            low[a]=min(low[a],dfsx[b]);
        }
    }
    if(low[a]==dfsx[a])
    {
        if(sta.back()!=a)return true;
        ++_scc;
        while(1)
        {
            int ls1=sta.back();
            sta.pop_back();
            scc[ls1]=_scc;
            if(ls1==a)
            {
                break;
            }
        }
    }
    return false;
}
bool judge(int x)
{
    clear();
    for(int i=1;i<=x;++i)
    {
        adde(a[i].a,a[i].b);
    }
    for(int i=1;i<=n;++i)
    {
```

```
        if(!dfsx[i])
        {
            if(dfs(i))return false;
        }
    }
    return true;
}
void solve()
{
    int l=1,r=m,ans=0;
    while(l<=r)
    {
        int m=(l+r)>>1;
        if(judge(m))l=m+1,ans=m;
        else r=m-1;
    }
    for(int i=1;i<=ans;++i)printf("1\n");
    for(int i=ans+1;i<=m;++i)printf("0\n");
    // printf("%d\n",ans);
}
int main()
{
    read();
    // printf("#%d#\n",judge(3));
    solve();
    return 0;
}
```

<题目跳转> <查看代码>