[ Tarjan 有向图强连通分量]poj 1904：King's Quest

<http://poj.org/problem?id=1904>

**题意：**

　　很久很久以前..

　　有一个国王..

　　他有好几个儿子..

　　这些王子都喜欢上了邻国的公主..

　　他们准备迎娶自己喜欢的公主中的一个..

　　国王就让宰相给列一个清单..

　　宰相就给了国王一个清单..上面写明了哪个王子将迎娶哪个邻国的公主..

　　但是调皮的国王不太满意~~

　　他还想知道他的儿子分别可以迎娶哪几个公主中的一个而不会让他的兄弟因此而吃醋..

　　所以宰相又得重新写清单了..

　　噢..对了~邻国的公主和王子人数一样多~<这是一个奇怪的国家..>

　　输入：

　　一个n表示有n个王子..

　　接下来n行每行有一个m表示第i个王子喜欢m个公主..

　　然后给出m个公主的序号..

　　最后一行有n个数..代表第i个王子最后迎娶了的公主..

　　输出：

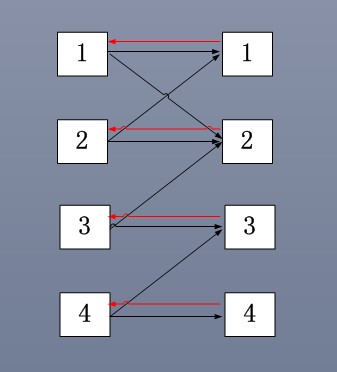
　　n行..每行第一个是m 表示有m个公主他可以选择~

　　然后m个数是公主的序号..

　　P.S 麻烦的国王还要求公主序号按从小到大给出..

<这是一些花心的王子..= =!! 可恨..>

**题意**：有n个女生和n个男生，给定一些关系表示男生喜欢女生(即两个人可以结婚)，再给定一个初始匹配,表示这个男生和哪个女生结婚，初始匹配必定是合法的。求每个男生可以和哪几个女生可以结婚且能与所有人不发生冲突(即确定某2个顶点连边后，其他2\*(n-1)个顶点的2分图是否还可以构成完美匹配)  
**分析**：想到匹配很正常，但是，如果想到在一个强连通分量里的点具有等价性，也许更好。  
**建图**：如果男i喜欢女j，连一条有向边i-->j；对于给定的初始化匹配，如果男i“分配”的是女j，连一条有向边j-->i（提示：i-->j已经连接）。具体见图示。现在要说明这样一个问题：和男i在同一个强连通分量的女j集合，为什么都是可选对象？原因很简单，在这个集合中，每个男i都喜欢所有的女生，也就是说每个男i对所有在该强连通分量中的女j都有一条直接相连的有向边。如果你不信，可以找个反例。。。   
**算法**：Tarjan，建图不是很难，关键在理解。

[](http://photo.blog.sina.com.cn/showpic.html#blogid=676070110100sx0t&url=http://s5.sinaimg.cn/orignal/67607011ga1bfcc6ff924)

**Tips：**

因为类似拆点了..所以数组要开大一点..

Cpp代码 收藏代码

#include<iostream>

#include<cstdio>

#include <algorithm>

#include<cstring>

using namespace std;

const int inf=1<<30;

const int nMax=30015;

const int mMax=500100;

class edge{

public:

int v,nex;

} e[mMax];

int k,head[nMax];//head[i]是以点i为起点的链表头部

void addedge(int a,int b){//向图中加边的算法，注意加上的是有向边//b为a的后续节点既是a---->b

e[k].v=b;

e[k].nex=head[a];

head[a]=k;k++;

}

int dfn[nMax],low[nMax],sta[nMax],top,atype,belon[nMax],dep; //atype 强连通分量的个数

bool insta[nMax];

void Tarjan(int u){ //我的Tarjan模版

int i,j;

dfn[u]=low[u]=++dep;

sta[++top]=u;

insta[u]=1;

for(i=head[u];i;i=e[i].nex){

int v=e[i].v;

if(!dfn[v]){

Tarjan(v);

low[u]=min(low[u],low[v]);

}

else{

if(insta[v])low[u]=min(low[u],dfn[v]);

}

}

if(dfn[u]==low[u]){

atype++; //强连通分量个数

do{

j=sta[top--];

belon[j]=atype; //第j个点属于第type个连通块

insta[j]=0;

}while(u!=j);

}

}

void init(){

k=1;

dep=1;

top=atype=0;

memset(insta,0,sizeof(insta)); //是否在栈中

memset(head,0,sizeof(head)); //静态链表头指针

memset(low,0,sizeof(low)); //Tarjan的low数组

memset(dfn,0,sizeof(dfn)); //Tarjan的dfn数组

memset(belon,0,sizeof(belon)); //记录每个点属于哪一个强连通分量

}

int ans[nMax];

int main(){

int n,a,m,i,j,b;

while(scanf("%d",&n)!=EOF){

init();

for(i=1;i<=n;i++){

scanf("%d",&m);

while(m--){

scanf("%d",&a);

addedge(i,n+a);

}

}

for(i=1;i<=n;i++){

scanf("%d",&a);

addedge(n+a,i);

}

for(i=1;i<=2\*n;i++){

if(!dfn[i]){

Tarjan(i);

}

}

for(i=1;i<=n;i++){

b=0;

for(j=head[i];j;j=e[j].nex){

int v=e[j].v;

if(belon[i]==belon[v]){

ans[b++]=v-n;

}

}

sort(ans,ans+b);

printf("%d ",b);

for(j=0;j<b;j++){

printf("%d ",ans[j]);

}

printf("\n");

}

}

return 0;

}