Hálózat központja

Minden számítógépes hálózat csomópontokból és bizonyos csomópontpárok között kiépített közvetlen kétirányú adatátvitelt biztosító kommunikációs vonalakból épül fel. A feladatban szereplő hálózatról tudjuk, hogy bármely két csomópont között pontosan egy olyan útvonal létezik, amely összeköti a két csomópontot. Adott p és q csomópont távolsága az a legkisebb k egész szám, amelyre létezik olyan $p = p_0, p_1, \ldots, p_k = q$ csomópontsorozat, hogy p_i és p_{i+1} $(i = 0, \ldots, k-1)$ között van kiépített közvetlen kommunikációs vonal. Minden csomópont fontos jellemzője az az érték, amely a többi csomóponttól vett távolság értékek maximuma. A hálózat központjának nevezzük azt a csomópontot, amelyre ez az érték a lehető legkisebb.

Feladat

Ijunk olyan programot, amely kiszámítja egy hálózat központját!

Bemenet

A standard bemenet első sora egy egész számot tartalmaz, a csomópontok n $(1 \leq n \leq 100000)$ számát. A további $n-1$ sor mindegyike olyan u v csomópontok sorszámát tartalmazza $(1 \leq u, v \leq n)$, amelyek között közvetlen kétirányú adatátviteli vonal van kiépítve. A bemenet teljesíti azt a feltételt, hogy bármely két csomópont között pontosan egy útvonal létezik.

Kimenet

A standard kimenet egyetlen sorába egy olyan csomópont sorszámát kell írni, amely a hálózat központja! Ha több ilyen lehet, akkor bármelyik megadható.

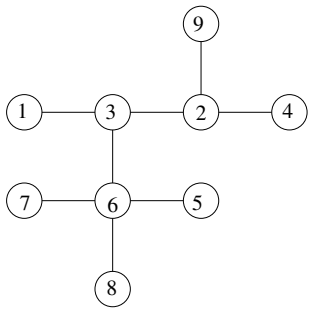
Példa

Bemenet

```
9
1 3
3 6
3 2
2 4
2 5
6 7
6 8
9 2
```

Kimenet

```
3
```



Korlátok

Időlimit: 0.1 mp.
Memórilimit: 8 MiB
Pontozás: a tesztesetek 40%-ában $n < 1000$

Megoldás

Tekintsük azt az irányítatlan $G = (V, E)$ gráfot, amelynek pontjai a hálózat csomópontjai, azaz $V = \{1, \ldots, n\}$, (u, v) akkor és csak akkor él a gráfban, ha az u és v között közvetlen kétirányú adatátviteli vonal van kiépítve.
Jelölje $Fok[p]$ a gráf p pontjának fokát, tehát azon q pontok számát, amelyre (p, q) él a gráfban.
Vegyük észre, hogy ha egyidejűleg kitöröljük a gráf összes olyan p pontját, amelyre $Fok[p] = 1$ (és természetesen az ilyen pontokhoz csatlakozó éleket is), akkor olyan gráfot kapunk, amelynek ugyanaz lesz a központja, mint

az eredetinek. Ismételjük ezt mindaddig, amíg egy, vagy 2 pont marad. A megmaradt pont (pontok) a hálózat központja. Az alábbi algoritmus egy lehetséges megoldása a problémának.

```
F1:=Üres
ciklus i=1-töl n-ig
    ha Fok[i]=1 akkor
        tegyük i-t az F1 halmazba
    elágazás vége
ciklus vége
hany:=n
ciklus amíg hany>2
    F2:=Üres
    ciklus F1 minden p elemére
        hany:=hany-1
        ciklus minden olyan q elemre, amelyre (p,q) él a gráfban
            Fok[q]:=Fok[q]-1
            ha Fok[q]=1 akkor
                tegyük q-t az F2 halmazba
            elágazás vége
        ciklus vége
    ciklus vége
    F1:=F2
ciklus vége
vegyük az F1 halmaz egy p elemét //F1-nek egy vagy két eleme van
kiír(p)
```

Vegyük észre, hogy az F1 és F2 halmazok helyett használhatunk egyetlen sort is ha F1-ből elem kivételt a Sorból, F2-be tételt pedig Sorba művelettel helyettesítjük. Ha a sort az F tömbbel és `eleje`, `vege` inedexekkel valósítjuk meg, akkor az algoritmus a következő lesz.

```
eleje:=1; vege:=0;
ciklus i=1-töl n-ig
    ha Fok[i]=1 akkor
        vege:=vege+1
        F[vege]:=i
    elágazás vége
ciklus vége
hany:=n
ciklus amíg hany>1
    p:=F[eleje] eleje:=eleje+1
    hany:=hany-1
    ciklus minden olyan q elemre, amelyre (p,q) él a gráfban
        Fok[q]:=Fok[q]-1
        ha Fok[q]=1 akkor
            vege:=vege+1
            F[vege]:=q
        elágazás vége
    ciklus vége
ciklus vége
p:=F[elso]
kiír(p)
```

Megvalósítás C++ nyelven

```
1  #include <iostream>
2  #include <vector>
3  #define maxN 10001
4  using namespace std;
5  vector<int> G[maxN];
6  int Fok[maxN];
7  int n;
8
9  void Beolvas(){
10    int p,q;
11    cin>>n;
12    for(int i=1;i<=n;i++) Fok[i]=0;
13    for (int i=1;i<n;i++){
14       cin>>p>>q;
15       G[p].push_back(q);
16       G[q].push_back(p);
17       Fok[p]++;
18       Fok[q]++;
19    }
20 }
21
22 int main(){
23    Beolvas();
24    int E[maxN];
25    int eleje=0, vege=0, hany=n, p;
26    for (int p=1;p<=n;p++)
27       if(Fok[p]==1)
28          E[vege++]=p;
29    while(hany>1){
30       p=E[eleje++];
31       hany--;
32       for (int q:G[p]){
33          Fok[q]--;
34          if(Fok[q]==1)
35             E[vege++]=q;
36       }
37    }
38    cout<<E[eleje]<<endl;
39    return 0;
40  }
```