Csapok nyitása

Egy csővezeték hálózatban a fő csőtől bármely másik csomópontba egyértelmű úton lehet eljutni. Minden cső (az első kivételével) két csomópontot köt össze, a csapot azzal a csomóponttal azonosítjuk, ahova a cső vége csatlakozik. Minden cső végén egy csap található, amely nyitható és zárható. Kezdetben minden csap zárva van, kivétel néhány – a csőhálózat végén –, amelyeken vizet szeretnénk kapni.

Készíts programot, amely megadja, hogy mely csapokat kell még kinyitnunk!

Bemenet

A *standard bemenet* első sorában a csapok száma van ($2 \le N \le 100\,000$). A következő N-1 sorban az egyes csövek kezdő- és végcsomópontja szerepel ($1 \le Kezd_i < Vég_i \le N$). A következő sorban azon csapok száma van ($1 \le Cs \le N$), amelyeken víznek kell kifolyni, amit a Cs ilyen csap sorszáma követ ($1 \le Csap_i \le N$).

Kimenet

A *standard kimenet* első sorába azon további csapok számát kell írni, amelyeknek nyitva kell lenni, hogy pontosan a bemenetben megadott Cs csapon keresztül kapjunk vizet! A második sorba ezen csapok sorszámai kerüljenek, növekvő sorrendben!

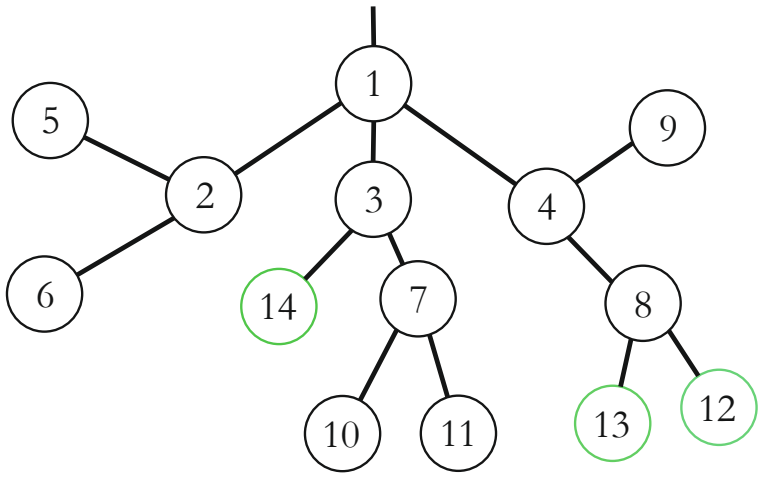
Példa

Bemenet

```
14
1 2
1 3
1 4
2 5
2 6
3 14
3 7
7 10
7 11
4 8
4 9
8 12
8 13
3 12 13 14
```

Kimenet

```
4
1 3 4 8
```



Korlátok

Időlimit: 0.2 mp.

Memórialimit: 32 MB

Repülés visszatéréssel

Egy országban egy nap néhány városból felszállt egy-egy repülő és mindegyik leszállt a kiinduló városuktól különböző helyen. A nap végén az égre nézve látjuk a csíkokat, amiket maguk után hagytak. Tudjuk, hogy mely városok között haladtak repülők, de azt nem, hogy mely irányokban.

Készíts programot, amely megadja, hogy hány olyan város van, ahova semmiképpen nem juthattunk vissza aznap!

Bemenet

A *standard bemenet* első sorában a városok száma ($1 \le N \le 10\,000$) és a csíkok száma ($1 \le M \le N$) van. A következő M sorban soronként egy-egy csík két végpontjának városindexe található ($1 \le A_i \ne B_i \le N$). A bemenetre teljesül, hogy legalább egy reptetés mindig lehetséges és minden városba legfeljebb 100 repülő érkezett.

Kimenet

A *standard kimenet* első sorába azon városok számát kell írni, amelyekbe semmiképpen nem juthattunk vissza aznap!

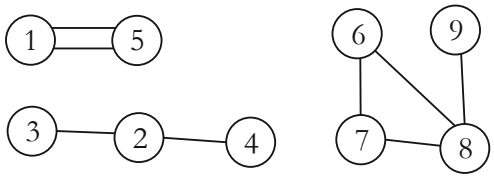
Példa

Bemenet

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

Kimenet

```
4
```



Magyarázat

1 és 5 között oda-vissza repülhetünk, 6-7-8 között körbe repülhetünk, így marad 4 város (2,3,4,9), vagy ahonnan nem indult gép vagy ahova nem érkezett gép:

Korlátok

Időlimit: 0.2 mp.

Memórialimit: 32 MB

Ütemezés azonos futási idővel

Egy 1 processzoros számítógép memóriájában programok vannak, mindegyik K időegységig használná a processzort. Az egyes programok a processzor idejéből 1-1 időegységet kapnak a memóriába kerülésük sorrendjében, az utolsónak bekerült után újra az első jön. Ha a program futási ideje lejárt, akkor kikerül a memóriából. Közben újabb programok kerülhetnek be a memóriába, az i. időpontban memóriába került program a futásra várók közül az utolsó helyre kerül!

Készíts programot, amely megadja, hogy a processzor időegységeit milyen sorrendben használják az egyes programok!

Bemenet

A *standard bemenet* első sorában a programok száma ($1 \le N \le 50\,000$) és az egyes programok futási ideje ($1 \le F \le 100$) van. A következő sorban az egyes programok kezdési ideje szerepel ($1 \le Kezd_i \le 1\,000\,000$), növekvő sorrendben.

Kimenet

A *standard kimenet* első sor i. száma az `i.` program befejeződési ideje legyen!

Példa

Bemenet	Kimenet
4 3 2 3 3 7	6 10 11 13

Magyarázat

A processzor idejét a programok a következőképpen használják:

1	2	3	4	5	6	7	8	9	10	11	12	13	14
—	1	1	2	3	1	2	3	4	2	3	4	4	—

Korlátok

Időlimit: 1.0 mp.

Memórialimit: 32 MB

Vasútvonal leghamarabb érkezéssel

Egy vasútvonal minden állomásáról ismerjük, hogy hova mikor indul vonat és az a célállomásra mikor érkezik. Útközben egyik vonat sem áll meg. Egy vonatról egy állomáson akkor szállhatunk át egy másik vonatra, ha az érkezési ideje kisebb, mint a másik vonat indulási ideje.

Készíts programot, amely megadja, hogy mikor érkezhetünk leghamarabb az utolsó állomásra az első állomásról indulva!

Bemenet

A *standard bemenet* első sorában az állomások száma ($1 \le N \le 1000$) és a vonatok száma ($1 \le M \le 100\,000$) található. A következő M sorban egy-egy vonat induló és végállomása ($1 \le A_i < B_i \le N$), valamint indulási és érkezési ideje ($1 \le Ind_i < Érk_i \le 1\,000\,000$) van, indulási idő szerint növekvő sorrendben.

Kimenet

A *standard kimenet* első sorába a legkorábbi időpontot kell írni, amikor az utolsó állomásra érhetünk az első állomásról indulva! Ha nincs megoldás, akkor -1-et kell kiírni!

Példa

Bemenet	Kimenet
4 8 1 3 5 18 1 2 7 10 3 4 10 15 2 4 12 29 1 2 15 19 2 3 15 19 2 4 20 25 3 4 22 28	25

Magyarázat

A második és a ötödik vonattal is indulhatunk, a második állomáson át kell szállni a hatodik vonatra, ami 25-re a végállomásra ér.

Korlátok

Időlimit: 0.2 mp.

Memórialimit: 32 MB