



Ezersziget

Ezersziget egy N gyönyörű szigetet tartalmazó szigetcsoport a Java-tengeren. A szigeteket 0-tól $N - 1$ -ig sorszámozzuk.

Van M kenu, melyeket 0-tól $M - 1$ -ig sorszámozzunk, és amelyekkel a szigetek között hajózhatunk. Minden i -re ($0 \leq i \leq M - 1$) az i . kenu az $U[i]$ vagy $V[i]$ szigeten van kikötve, és csak az $U[i]$ és $V[i]$ sziget között hajózhatunk vele. Pontosabban, ha a kenu az $U[i]$ szigeten van kikötve, akkor az $U[i]$ szigetről a $V[i]$ szigetre hajózhatunk vele, ami után a kenu a $V[i]$ szigeten lesz kikötve. Hasonlóképpen, ha a kenu a $V[i]$ szigeten van kikötve, akkor a $V[i]$ szigetről az $U[i]$ szigetre hajózhatunk vele, ami után a kenu az $U[i]$ szigeten lesz kikötve. Kezdetben a kenu az $U[i]$ szigeten van kikötve. Akár több kenuval is hajózhatunk ugyanazon két sziget között. Szintén lehetséges, hogy ugyanazon a szigeten több kenu van kikötve.

Biztonsági okokból a kenukat minden út után elő kell készíteni a következő hajózáshoz, ezért ugyanazt a kenut nem lehet kétszer egymás után használni. Azaz, hogyha az i . kenuval hajóztunk, akkor közvetlenül utána egy másik kenut kell használni.

Bu Dengklek utazást szervez a szigetek között. Az utazása akkor és csak akkor **szabályos**, ha a következő három feltétel mindegyike teljesül:

- A 0 sorszámú szigetről indul és oda is tér vissza.
- Az útja során legalább egy 0-tól különböző szigetet érint.
- Az útja végeztével minden kenunak az utazás előtti szigeten kell lennie. Vagyis minden i -re ($0 \leq i \leq M - 1$) az i . kenu az $U[i]$ szigeten legyen.

Segíts Bu Dengkleknek legfeljebb 2 000 000 hajózást tartalmazó szabályos utazás megtalálásában, vagy jelezd, ha ilyen szabályos utazás nem lehetséges. Bizonyítható, hogy a feladatban adott feltételek teljesülése esetén (ld. Feltételek fejezet), ha létezik szabályos utazás, akkor olyan is van, ami legfeljebb 2 000 000 hajózást tartalmaz.

Megvalósítás

A következő függvényt kell megvalósítanod:

```
union(bool, int[]) find_journey(int N, int M, int[] U, int[] V)
```

- N : a szigetek száma.

- M : kenek száma.
- U, V : a kenu kezdeti helyzetét megadó M elemű tömbök.
- A függvénynek vagy egy logikai értéket, vagy egészek tömbjét kell visszaadnia. (Implementációs részleteket ld. a Tudnivalók lapon.)
 - Ha nem létezik szabályos utazás, akkor a `false` értéket kell visszaadnia.
 - Ha létezik szabályos utazás, két lehetőség van:
 - A teljes pontszám eléréséhez, a függvények egy legfeljebb 2 000 000 egész számot tartalmazó tömböt kell visszaadni, amely egy szabályos utazást ír le. E tömb elemei az utazás során használt kenu sorszámai legyenek a hajózás sorrendjében.
 - Részpontszámot kapsz, ha a függvény `true`-val, vagy több mint 2 000 000 elemű egészek tömbjével, vagy egy nem szabályos utazást leíró egészek tömbjével tér vissza, akkor a megoldásra részpontszámot adunk (ld. Részfeladatok fejezetet további részletekért).
- A függvényt pontosan egyszer hívják meg.

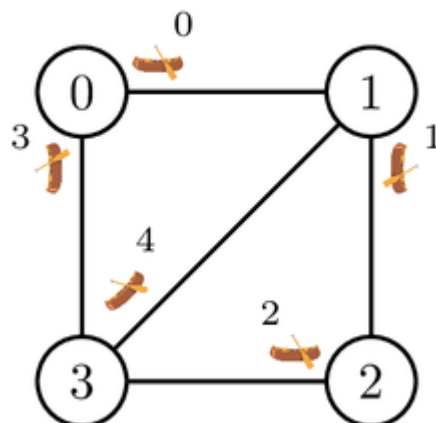
Példa

1. példa

Tekintsük a következő függvényhívást:

```
find_journey(4, 5, [0, 1, 2, 0, 3], [1, 2, 3, 3, 1])
```

A szigetek és kenuk az alábbi képen láthatók.



Egy lehetséges szabályos utazás a következő. Bu Dengklek először sorrendben a 0., 1., 2. és 4. kenukkal hajózik. Ekkor az 1. szigeten van. Ezután, Bu Dengklek a 0. kenu újra használhatja, hiszen az az 1. szigeten van kikötve, és nem ez volt az utoljára használt kenu. A 0. kenuval a 0. szigetre érkezik. Azonban az 1., 2., 4. kenu nem a kezdeti helyén van, így Bu Dengklek folytatja utazását

sorban a 3., 2., 1. és 3. kenuval. Ekkor Bu Denglek visszatér a 0. szigetre és az összes kenu a kiindulási helyére kerül.

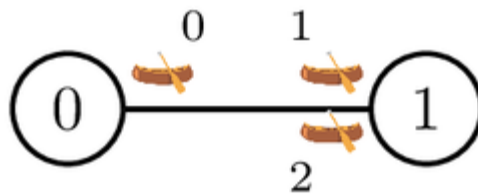
Tehát a $[0, 1, 2, 4, 0, 3, 2, 1, 4, 3]$ tömb visszaadása egy szabályos utazást ad meg.

2. példa

Tekintsük a következő függvényhívást:

```
find_journey(2, 3, [0, 1, 1], [1, 0, 0])
```

A szigetek és kenuk az alábbi képen láthatók.



Bu Denglek csak a 0. kenuval kezdhet, ami után csak az 1.-vel vagy 2.-kal folytathatja, mert a 0. kenut nem használhatja kétszer egymás után. Mindkét esetben a 0. szigetre kerül. A kenuk azonban nem a kiindulási helyükön vannak és Bu Denglek nem használhatja egyik kenut sem a továbbiakban, mivel a 0. szigeten csak az a kenu van, amivel oda érkezett. Mivel nincs szabályos utazás, ezért a függvény `false` értékkel tér vissza.

Korlátok

- $2 \leq N \leq 100\,000$
- $1 \leq M \leq 200\,000$
- $0 \leq U[i] \leq N - 1$ és $0 \leq V[i] \leq N - 1$ (minden i -re, ami $0 \leq i \leq M - 1$)
- $U[i] \neq V[i]$ (minden i -re, ami $0 \leq i \leq M - 1$)

Részfeladatok

1. (5 pont) $N = 2$
2. (5 pont) $N \leq 400$. Minden x és y szigetpárra ($0 \leq x < y \leq N - 1$) pontosan egy kenu van az x . szigeten és pontosan egy kenu van az y . szigeten.
3. (21 pont) $N \leq 1000$, M páros, és minden **páros** i -re ($0 \leq i \leq M - 1$) mind az i ., mind pedig az $i + 1$. kenuval az $U[i]$. és $V[i]$. szigetek között hajózhatunk. Az i . kenu az $U[i]$ szigeten, az $i + 1$. kenu a $V[i]$. szigeten van. Formálisan $U[i] = V[i + 1]$ és $V[i] = U[i + 1]$.

4. (24 pont) $N \leq 1000$, M páros, és minden **páros** i -re ($0 \leq i \leq M - 1$) mind az i ., mind pedig az $i + 1$. kenuval az $U[i]$. és $V[i]$. szigetek között hajózhatunk. Mindkét kenu az $U[i]$ szigeten van. Formálisan $U[i] = U[i + 1]$ és $V[i] = V[i + 1]$.
5. (45 pont) Nincs további feltétel.

Minden olyan tesztesetnél, ahol létezik szabályos utazás, a megoldásodat a következőképpen pontozzák:

- teljes pontszám jár, ha egy szabályos utazást adsz meg,
- a pontok 35%-át kapod, ha vagy `true` értékkel, vagy 2 000 000-nál nagyobb elemszámú tömbbel térsz vissza, vagy ha a tömb nem szabályos utazást ír le,
- 0 pontot minden más esetben.

Minden olyan tesztesetnél, ahol nem létezik szabályos utazás, a megoldásodat a következőképpen pontozzák:

- teljes pontszám jár, ha `false` értéket adsz vissza,
- 0 pontot egyébként.

Megjegyzendő, hogy a részfeladatra kapott pontszám a részfeladat teszteseteire kapott pontszámok minimuma.

Mintaértékelő

A mintaértékelő a standard bemenetről a következő formában olvas be:

- 1. sor: $N M$
- $2 + i$. sor ($0 \leq i \leq M - 1$): $U[i] V[i]$

A mintaértékelő a választ a következő formában írja ki a standard kimenetre:

- Ha `find_journey` logikai értékkel tér vissza:
 - 1. sor: 0
 - 2. sor: 0, ha `find_journey` `false` értékkel, 1, ha `true` értékkel tér vissza.
- Ha `find_journey` egészek tömbjével tér vissza, jelezzük e tömb elemeit sorban $c[0], c[1], \dots, c[k - 1]$ jelöléssel. A mintaértékelő ezt írja ki:
 - 1. sor: 1
 - 2. sor: k
 - 3. sor: $c[0] c[1] \dots c[k - 1]$