Highway Tolls

U Japanu su gradovi povezani mrežom autoputeva. Mreža se sastoji od N gradova i M autoputeva. Svaki autoput povezuje par različitih gradova. Ne postoje dva autoputa koja povezuju isti par gradova. Gradovi su označeni od 0 do $N-1$, dok su autoputevi označeni od 0 do $M-1$. Autoputevi su dvosmjerni i moguće je iz bilo kojeg od gradova stići do bilo kojeg drugog koristeći autoputeve.

Putarina se naplaćuje za vožnju na svakom autoputu. Cijena putarine zavisi od **gustine saobraćaja** na autoputu. Saobraćaj može biti **rijedak** ili **gust**. Kada je saobraćaj rijedak, cijena putarina je A jena (ako još niste naučili to je japanska valuta:). Kada je saobraćaj gust, cijena putarine je B jena. Garantovano je da je $A < B$. Uzmite u obzir da su vam vrijedosti A i B poznate.

Imate uređaj koji, kada mu je poznato stanje saobraćaja na svim autoputevima, računa najmanju ukupnu putarinu koju bi neko morao platiti da bi putovao između gradova S i T. ($S \neq T$) pod tim stanjem saobraćaja.

Međutim, ovaj uređaj je samo prototip. Vrijednosti S i T su fiksirane (tj. hardkodirani u mašini) i vama nisu poznate. Želite saznati S i T. Da biste to uradili, planirate postaviti mašini nekoliko stanja saobraćaja i iskoristiti dobijene vrijednosti putarine da zaključite vrijednosti S i T. S obzirom da je davanje stanja u saobraćaju mašini skupo, ne želite da koristite mašinu previše puta.

Detalji implementacije

Trebate implementirati sljedeću funkciju:

```
find_pair(int N, int[] U, int[] V, int A, int B)
```

- `N`: broj gradova.
- `U` i `V`: nizovi dužine M, gdje je M broj autoputeva koji povezuju gradove. Za svako i ($0 \leq i \leq M-1$), autoput i povezuje gradove `U[i]` i `V[i]`.
- `A`: cijena putarine po autoputu kada je saobraćaj rijedak.
- `B`: cijena putarine po autoputu kada je saobraćaj gust.
- Funkcija će biti pozvana samo jednom po testnom slučaju.
- Uzmite u obzir da je vrijednost M dužina nizova, i može biti određena na način prikazan u obavještenju o implementaciji.

Funkcija `find_pair` može pozvati sljedeću funkciju:

```
int64 ask(int[] w)
```

- Dužina `w` mora biti M. Niz `w` opisuje stanje saobraćaja.
- Za svako i ($0 \leq i \leq M-1$), `w[i]` daje stanje saobraćaja na autoputu i. Vrijednost `w[i]` mora biti 0 ili 1.
  - `w[i] = 0` znači da je saobraćaj na autoputu i rijedak.
  - `w[i] = 1` znači da je saobraćaj na autoputu i gust.
- Ova funkcija vraća najmanju ukupnu cijenu putarine za putovanje između gradova S i T, pod saobraćajnim uslovima navedenim pod `w`.
- Ova funkcija ne može biti pozvana više od 100 puta (po testnom slučaju).

`find_pair` treba pozvati sljedeću funkciju za prijavu odgovora:

```
answer(int s, int t)
```

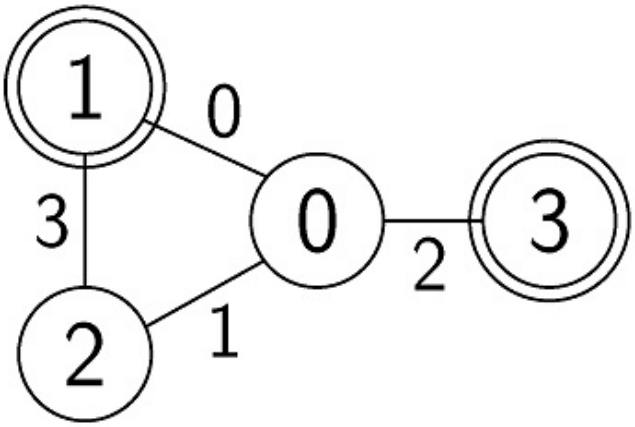
- `s` i `t` moraju biti par S i T (pri čemu redosljed nije bitan).
- Ova funkcija mora biti pozvana tačno jednom.

Ako neki od uslova iznad nije zadovoljen, vaš program je ocijenjen kao **`Wrong Answer`**. U suprotnom, vaš program je ocijenjen kao **`Accepted`** i vaši bodovi se računaju prema broju poziva `ask` (vidjeti sekciju Podzadaci).

Primjer

Neka su $N = 4$, $M = 4$, $U = [0, 0, 0, 1]$, $V = [1, 2, 3, 2]$, $A = 1$, $B = 3$, $S = 1$ i $T = 3$.

Program za ocjenjivanje (grader) poziva `find_pair(4, [0, 0, 0, 1], [1, 2, 3, 2], 1, 3)`.



Na slici iznad grana sa brojem i odgovara autoputu i. Neki od mogućih poziva `ask` i odgovarajuće povratne vrijednosti su napisane ispod:

Call	Return
`ask([0, 0, 0, 0])`	2
`ask([0, 1, 1, 0])`	4
`ask([1, 0, 1, 0])`	5
`ask([1, 1, 1, 1])`	6

Za poziv funkcije `ask([0, 0, 0, 0])`, saobraćaj na svakom od autoputeva je rijedak i cijena putarine je 1. Najjeftinija ruta od $S = 1$ do $T = 3$ je $1 \to 0 \to 3$. Ukupna cijena putarine za ovu rutu je 2. Tako da funkcija vraća 2.

Za ispravan odgovor, procedura `find_pair` treba pozvati `answer(1, 3)` ili `answer(3, 1)`.

Fajl `sample-01-in.txt` u zippovanom prilogu odgovara ovom primjeru. Drugi primjeri ulaza se također nalaze u prilogu.

U zip-fajlu uz ovaj zadatak, fajl `sample-01-in.txt` odgovara ovom primjeru. Zip-fajl sadrži i druge primjere.

Ograničenja

- $2 \leq N \leq 90\,000$
- $1 \leq M \leq 130\,000$
- $1 \leq A < B \leq 1\,000\,000\,000$
- Za svako $0 \leq i \leq M - 1$
  - $0 \leq U[i] \leq N - 1$
  - $0 \leq V[i] \leq N - 1$
  - $U[i] \neq V[i]$
- $(U[i], V[i]) \neq (U[j], V[j])$ and $(U[i], V[i]) \neq (V[j], U[j])$ $(0 \leq i < j \leq M - 1)$
- Možete stići iz bilo kojeg grada u bilo koji drugi koristeći autoputeve.
- $0 \leq S \leq N - 1$
- $0 \leq T \leq N - 1$
- $S \neq T$

U ovom zadatku, program za ocjenjivanje (grader) NIJE adaptivan. Drugim riječima, S i T su fiksirani na početku pokretanja program za ocjenjivanje i ne zavise od upita koje postavi vaše rješenje.

Podzadaci

1. (5 bodova) jedan od S ili T je 0, $N \leq 100$, $M = N - 1$
2. (7 bodova) jedan od S ili T je 0, $M = N - 1$
3. (6 bodova) $M = N - 1$, $U[i] = i$, $V[i] = i + 1$ $(0 \leq i \leq M - 1)$

4. (33 boda) $M = N - 1$
5. (18 bodova) $A = 1$, $B = 2$
6. (31 bod) Bez dodatnih ograničenja

Pretpostavimo da je vaš program ocijenjen kao **`Accepted`**, i poziva funkciju `ask` X puta. U tom slučaju se vaš broj bodova P za taj testni slučaj, u zavisnosti od broja podzadatka, računa na sljedeći način:

- Podzadatak 1. $P = 5$.
- Podzadatak 2. ako je $X \leq 60$, $P = 7$. U suprotnom $P = 0$.
- Podzadatak 3. ako je $X \leq 60$, $P = 6$. U suprotnom $P = 0$.
- Podzadatak 4. ako je $X \leq 60$, $P = 33$. U suprotnom $P = 0$.
- Podzadatak 5. ako je $X \leq 52$, $P = 18$. U suprotnom $P = 0$.
- Podzadatak 6.
  - Ako je $X \leq 50$, onda je $P = 31$.
  - Ako je $51 \leq X \leq 52$, onda je $P = 21$.
  - Ako je $53 \leq X$, onda je $P = 0$.

Uzmite u obzir da je vaš broj bodova za svaki podzadatak jednak najmanjem broju bodova za testne slučajeve u tom podzadatku.

Primjer programa za ocjenjivanje (sample grader)

Programa za ocjenjivanje učitava ulaz u sljedećem formatu:

- red 1: N M A B S T
- red $2 + i$ ($0 \leq i \leq M - 1$): $U[i]$ $V[i]$

Ako je vaš program ocijenjen kao **`Accepted`**, program za ocjenjivanje (grader) ispisuje `Accepted: q`, gdje je `q` broj poziva funkcije `ask`.

Ako je vaš program ocijenjen **`Wrong Answer`**, program za ocjenjivanje (grader)štampa `Wrong Answer: PORUKA`, gdje je `PORUKA` jedna od sljedećih:

- `answered not exactly once`: Funkcija `answer` nije pozvana tačno jednom.
- `w is invalid`: Dužina `w` koji je poslat u `ask` nije jednaka M ili `w[i]` nije niti 0 niti 1 za neko i ($0 \leq i \leq M - 1$).
- `more than 100 calls to ask`: Funkcija `ask` je pozvana više od 100 puta.
- `{s, t} is wrong`: Funkcija `answer` je pozvana sa neispravnim parom `s` i `t`.