# Highway Tolls

În Japonia, orașele sunt conectate de o rețea de autostrăzi. Rețeaua este formată din $N$ orașe și $M$ autostrăzi. Fiecare autostradă conectează o pereche de orașe distincte. Nu există două autostrăzi distincte care conectează aceeași pereche de orașe. Orașele sunt numerotate de la $0$ la $N-1$ și autostrăzile sunt numerotate de la $0$ la $M-1$. Puteți conduce pe oricare autostradă în ambele direcții. Puteți călători, folosind autostrăzile, între oricare două orașe.

O taxă este percepută pentru conducerea pe fiecare din autostrăzi. Taxa pe autostradă depinde de condițiile de **trafic**. Traficul poate fi **relaxat** sau **intens**. Când traficul este relaxat taxa este de $A$ yen (valută japoneză). Când traficul este intens taxa este de $B$ yen. Se garantează că $A < B$. Luați la cunoștință că valorile $A$ și $B$ sunt cunoscute.

Aveți un dispozitiv, care, pentru condiții date ale traficului pe toate autostrăzile calculează taxa totală minimă pe care cineva trebuie să o achitate pentru a călători între orașele $S$ și $T$ ($S \neq T$), în condiții de trafic specificate.

Totuși, dispozitivul este doar un prototip. Valorile $S$ și $T$ sunt fixate (adică în echipament) și necunoscute. Trebuie să determinați valorile $S$ și $T$. Pentru a realiza aceasta, planificați să specificați dispozitivului anumite condiții de trafic și să folosiți valorile taxelor calculate de acesta pentru a deduce $S$ și $T$. Deoarece specificarea condițiilor de trafic costă, nu doriți să folosiți dispozitivul de multe ori.

## Detalii de Implementare

Trebuie să implementați următoarea procedură:

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

- `N`: numărul de orașe.
- `U` și `V`: tablouri unidimensionale cu $M$ elemente, unde $M$ este numărul de autostrăzi care conectează orașele. Pentru fiecare $i$ ($0 \leq i \leq M-1$), autostrada $i$ conectează orașele `U[i]` și `V[i]`.
- `A`: taxa de autostradă când traficul este relaxat.
- `B`: taxa de autostradă când traficul este intens.
- Această procedură este apelată exact o dată pentru fiecare test.
- Luați la cunoștință că valoarea $M$ reprezintă dimensiunea tablourilor și poate fi obținută după cum este indicat în Observațiile de implementare.

Procedura `find_pair` poate apela următoarea funcție:

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

- Dimensiunea lui `w` trebuie să fie $M$. Tabloul `w` descrie condițiile de trafic.
- Pentru fiecare $i$ ($0 \leq i \leq M-1$), `w[i]` descrie condițiile de trafic pe autostrada $i$. Valoarea `w[i]` trebuie să fie 0 sau 1.
  - `w[i] = 0` înseamnă că traficul pe autostrada $i$ este relaxat.
  - `w[i] = 1` înseamnă că traficul pe autostrada $i$ este intens.
- Această funcție întoarce taxa totală minimă pentru călătoria între orașele $S$ și $T$, în condițiile de trafic specificate de `w`.
- Această funcție poate fi apelată de cel mult 100 de ori (pentru fiecare test).

`find_pair` trebuie să apeleze următoarea procedură pentru întoarce răspunsul:

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

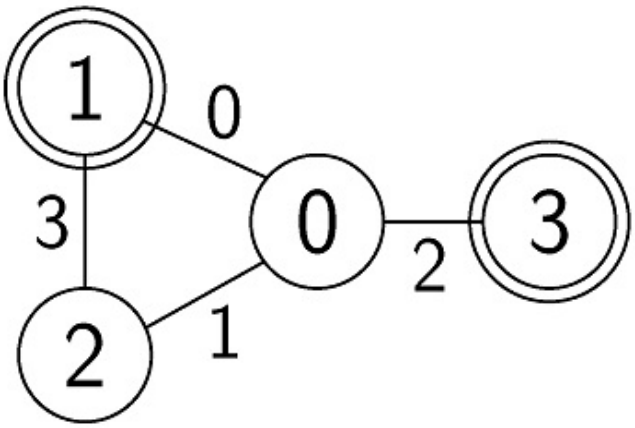
- `s` și `t` trebuie să fie perechea $S$ și $T$ (ordinea nu contează).
- Această procedură trebuie să fie apelată exact o dată.

Dacă vreuna din condițiile descrise mai sus nu este respectată, programul este evaluat cu **`Wrong Answer`**. În caz contrar programul este evaluat cu **`Accepted`** și punctajul este calculat după numărul de apeluri ale lui `ask` (vedeți Subtask-uri).

# Exemplu

Fie $N = 4$, $M = 4$, $U = [0, 0, 0, 1]$, $V = [1, 2, 3, 2]$, $A = 1$, $B = 3$, $S = 1$, și $T = 3$.

Grader-ul apelează `find_pair(4, [0, 0, 0, 1], [1, 2, 3, 2], 1, 3)`.




În figura de mai sus, muchia cu numărul $i$ corespunde autostrăzii $i$. Unele apeluri posibile către `ask` și valorile corespunzătoare întoarse sunt descrise mai jos:

| Apel | Valoare întoarsă |
|---|---|
| `ask([0, 0, 0, 0])` | 2 |
| `ask([0, 1, 1, 0])` | 4 |
| `ask([1, 0, 1, 0])` | 5 |
| `ask([1, 1, 1, 1])` | 6 |

La apelul funcției `ask([0, 0, 0, 0])`, traficul pe fiecare autostradă este relaxat și taxa de autostradă este 1. Ruta de cost minim de la $S = 1$ la $T = 3$ este $1 \to 0 \to 3$. Taxa totală pentru această rută este 2. Astfel, funcția întoarce 2.

Pentru un răspuns corect, procedura `find_pair` ar trebui să apeleze `answer(1, 3)` sau `answer(3, 1)`.

Fișierul `sample-01-in.txt` din pachetul arhivat anexat corespunde acestui exemplu. Alte exemple sunt disponibile în același pachet.

## Restricții

- $2 \leq N \leq 90\,000$
- $1 \leq M \leq 130\,000$
- $1 \leq A < B \leq 1\,000\,000\,000$
- Pentru fiecare $0 \leq i \leq M - 1$
  - $0 \leq U[i] \leq N - 1$
  - $0 \leq V[i] \leq N - 1$
  - $U[i] \neq V[i]$
- $(U[i], V[i]) \neq (U[j], V[j])$ and $(U[i], V[i]) \neq (V[j], U[j])$ $(0 \leq i < j \leq M - 1)$
- Puteți călători de la orice oraș la altul folosind autostrăzile.
- $0 \leq S \leq N - 1$
- $0 \leq T \leq N - 1$
- $S \neq T$

În această problemă, grader-ul NU este adaptiv. Aceasta înseamnă că $S$ și $T$ sunt fixate la începutul rulării grader-ului și nu vor depinde de întrebările puse de soluția dumneavoastră.

## Subtask-uri

1. (5 puncte) unul dintre orașele $S$ sau $T$ este 0, $N \leq 100$, $M = N - 1$
2. (7 puncte) unul dintre orașele $S$ sau $T$ este 0, $M = N - 1$
3. (6 puncte) $M = N - 1$, $U[i] = i$, $V[i] = i + 1$ $(0 \leq i \leq M - 1)$
4. (33 puncte) $M = N - 1$
5. (18 puncte) $A = 1$, $B = 2$

6. (31 puncte) Fără constrângeri adiționale

Să presupunem că programul dumneavoastră a fost evaluat ca **Accepted**, și apelează `ask` de $X$ ori. Pentru acest test punctajul $P$, în funcție de numărul subtask-ului, este calculat după cum urmează:

- Subtask 1. $P = 5$.
- Subtask 2. Dacă $X \leq 60$, $P = 7$. Altfel $P = 0$.
- Subtask 3. Dacă $X \leq 60$, $P = 6$. Altfel $P = 0$.
- Subtask 4. Dacă $X \leq 60$, $P = 33$. Altfel $P = 0$.
- Subtask 5. Dacă $X \leq 52$, $P = 18$. Altfel $P = 0$.
- Subtask 6.
  - Dacă $X \leq 50$, $P = 31$.
  - Dacă $51 \leq X \leq 52$, $P = 21$.
  - Dacă $53 \leq X$, $P = 0$.

Luați la cunoștință că punctajul pentru fiecare subtask este minimul punctajelor obținute pe testele din acel subtask.

# Grader local

Grader-ul local citește detele de intrare în următorea formă:

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

Dacă programul dumneavoastră a fost evaluat ca **Accepted**, grader-ul local va afișa `Accepted: q`, unde `q` este numărul de apeluri ale lui `ask`.

Dacă programul dumneavoastră a fost evaluat ca **Wrong Answer**, grader-ul local va afișa `Wrong Answer: MSG`, unde `MSG` este unul dintre mesajele:

- `answered not exactly once`: Procedura `answer` nu a fost apelată exact o dată.
- `w is invalid`: Dimensiunea parametrului `w` al funcției `ask` nu este $M$ sau `w[i]` nu este nici 0 nici 1 pentru unele valori ale lui $i$ ($0 \leq i \leq M - 1$).
- `more than 100 calls to ask`: Funcția `ask` a fost apelată mai mult de 100 ori.
- `{s, t} is wrong`: Procedura `answer` este apelată cu o pereche incorectă `s` și `t`.