Highway Tolls

In Japan zijn steden verbonden door een netwerk van snelwegen. Dit netwerk bestaat uit N steden en M snelwegen. Elke snelweg verbindt een paar verschillende steden. Er zijn geen twee snelwegen die hetzelfde tweetal steden verbindt. Steden zijn genummerd van 0 tot en met $N-1$, de snelwegen zijn genummerd van 0 tot en met $M-1$. Je kan een snelweg in beide richtingen berijden. Je kunt van elke stad naar elke andere stad reizen over de snelwegen.

Er wordt tol geheven voor het gebruik van een snelweg. De tol hangt af van de **drukte** op de snelweg. Er is ofwel **weinig** of **veel** verkeer. Bij weinig verkeer is de tol A yen (Japanse valuta), bij veel verkeer is de tol B yen. Het is gegarandeerd dat $A < B$. Merk op dat je de waarden van A en B weet.

Je heb een machine die, gegeven de druktes op alle snelwegen, de minimale totale tol berekent die iemand moet betalen om tussen de steden S en T ($S \neq T$) te reizen bij deze verkeerssituatie.

De machine is slechts een prototype. De waarden van S en T staan vast (dus ze zijn gehardcoded in de machine) en zijn niet aan jou bekend. Je wilt S en T graag bepalen. Om dit te doen ben je van plan verschillende verkeerssituaties aan de machine voor te leggen. Omdat het duur is om de druktes in te voeren wil je de machine niet vaak gebruiken.

Implementatiedetails

Implementeer de volgende procedure:

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

- `N`: het aantal steden.
- `U` en `V`: arrays van lengte M, waar M het aantal snelwegen is. Voor elke i ($0 \leq i \leq M-1$), verbindt de snelweg i de steden `U[i]` en `V[i]`.
- `A`: de tol bij weinig verkeer.
- `B`: de tol bij veel verkeer.
- Deze procedure wordt precies één keer aangeroepen voor elke test.
- Merk op dat de waarde van M de lengte is van de beide arrays, die verkregen kan worden zoals aangegeven in de implementatieopmerkingen.

De procedure `find_pair` kan de volgende functie aanroepen:

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

- De lengte van `w` moet gelijk zijn aan M. De array `w` beschrijft de verkeerssituatie.
- Voor elke i ($0 \leq i \leq M-1$), geeft `w[i]` de drukte op snelweg i. De waarde van `w[i]` moet ofwel 0 of 1 zijn.
  - `w[i] = 0` betekent dat er weinig verkeer is op snelweg i.
  - `w[i] = 1` betekent dat er veel verkeer is op snelweg i.
- De functie geeft de minimale totale tol terug voor het reizen tussen de steden S en T bij de verkeerssituatie gegeven door `w`.
- Deze functie kan maximaal 100 keer worden aangeroepen (voor elke test).

`find_pair` kan de volgende functie aanroepen om het antwoord terug te geven:

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

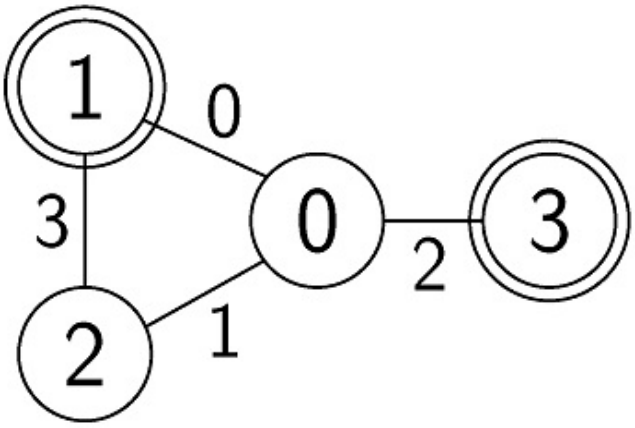
- `s` en `t` moet het paar S en T zijn (de volgorde maakt niet uit)
- Deze procedure moet precies één keer worden aangeroepen.

Als aan sommige van de hierboven gegeven voorwaarden niet is voldaan wordt je inzending beoordeeld als **`Wrong Answer`**. Anders wordt je inzending beoordeeld als **`Accepted`** en wordt je score berekend op basis van het aantal aanroepen van `ask` (zie Subtaken).

Voorbeeld

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

De grader roept `find_pair(4, [0, 0, 0, 1], [1, 2, 3, 2], 1, 3)` aan.



In de afbeelding hierboven komt de verbinding met nummer i overeen met snelweg i. Enkele mogelijke aanroepen van `ask` met de overeenkomende waarde die wordt

teruggegeven zijn hieronder gegeven:

Aanroep	Returnwaarde
`ask([0, 0, 0, 0])`	2
`ask([0, 1, 1, 0])`	4
`ask([1, 0, 1, 0])`	5
`ask([1, 1, 1, 1])`	6

Voor de aanroep `ask([0, 0, 0, 0])`, is er weinig verkeer op elke snelweg en is de tol voor elke snelweg 1. De goedkoopste route van $S = 1$ tot $T = 3$ is $1 \to 0 \to 3$. De totale tol voor deze route is 2. Dus de functie geeft 2 terug.

Voor het correct antwoord moet de procedure `find_pair` ofwel `answer(1, 3)` of `answer(3, 1)` aanroepen.

Het bestand `sample-01-in.txt` in de gezipte bijlage komt overeen met dit voorbeeld. De andere voorbeeldinvoers zijn ook beschikbaar in de bijlage.

Randvoorwaarden

- $2 \leq N \leq 90\,000$
- $1 \leq M \leq 130\,000$
- $1 \leq A < B \leq 1\,000\,000\,000$
- Voor elke $0 \leq i \leq M - 1$
  - $0 \leq U[i] \leq N - 1$
  - $0 \leq V[i] \leq N - 1$
  - $U[i] \neq V[i]$
- $(U[i], V[i]) \neq (U[j], V[j])$ en $(U[i], V[i]) \neq (V[j], U[j])$ $(0 \leq i < j \leq M - 1)$
- Je kunt reizen van elke stad naar elke andere stad over snelwegen.
- $0 \leq S \leq N - 1$
- $0 \leq T \leq N - 1$
- $S \neq T$

In dit probleem is the grader NIET adaptief. Dit betekent dat S en T vast staan voor het starten van de grader en dat deze niet afhangen van de vragen die door jouw programma worden gesteld.

Subtasks

1. (5 punten) S of T is 0, $N \leq 100$, $M = N - 1$
2. (7 punten) S of T is 0, $M = N - 1$
3. (6 punten) $M = N - 1$, $U[i] = i$, $V[i] = i + 1$ $(0 \leq i \leq M - 1)$

4. (33 punten) $M = N - 1$
5. (18 punten) $A = 1$, $B = 2$
6. (31 punten) Geen aanvullende voorwaarden

Indien je inzending is beoordeeld als **Accepted**, en X aanroepen doet aan `ask`, dan wordt je score P voor de test, afhankelijk van het subtaaknummer, als volgt beoordeeld:

- Subtaak 1. $P = 5$.
- Subtaak 2. If $X \leq 60$, $P = 7$. Anders $P = 0$.
- Subtaak 3. If $X \leq 60$, $P = 6$. Anders $P = 0$.
- Subtaak 4. If $X \leq 60$, $P = 33$. Anders $P = 0$.
- Subtaak 5. If $X \leq 52$, $P = 18$. Anders $P = 0$.
- Subtaak 6.
  - Als $X \leq 50$, $P = 31$.
  - Als $51 \leq X \leq 52$, $P = 21$.
  - Als $53 \leq X$, $P = 0$.

Merk op dat je score voor elke subtaak gelijk is aan het minimum van de scores voor alle tests in de subtaak.

Voorbeeldgrader

De voorbeeldgrader leest de invoer in het volgende format:

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

Als je programma wordt beoordeeld als **Accepted**, drukt de sample grader `Accepted: q` af, met `q` het aantal aanroepen aan `ask`.

Als je programma wordt beoordeeld als **Wrong Answer**, print het `Wrong Answer: MSG`, waar `MSG` een van de volgende is:

- `answered not exactly once`: De procedure `answer` was niet precies één keer aangeroepen.
- `w is invalid`: De lengte van `w` gegeven aan `ask` is niet M of `w[i]` is niet 0 of 1 voor sommige i ($0 \leq i \leq M - 1$).
- `more than 100 calls to ask`: De function `ask` is meer dan 100 keer aangeroepen.
- `{s, t} is wrong`: De procedure `answer` is aangeroepen met een verkeerd paar `s` en `t`.