高速公路收費

日本的城市是由一個高速公路網絡把很多城市連接起來。假設這個網絡包含 N 個城市和 M 條高速公路。每條高速公路都連接著二個不同的城市。而沒有二條高速公路會連接相同的二個城市。城市的編號是從 0 到 $N-1$，高速公路的編號則是從 0 到 $M-1$。你能夠任意在一條高速公路上雙向行駛。從任何一個城市出發你也能夠使用這些高速公路到達另外任何一個城市。

使用每條高速高路都需要收費。每條高速公路的收費都會依據它的**交通情況**來判斷，我們把交通情況分為**順暢**和**繁忙**二種。當一條高速公路的交通情況為順暢時，費用為 A 円（日本貨幣），而當交通情況為繁忙時，費用為 B 円。可以肯定 $A < B$。 注意你是早已知道 A 和 B 的值。

你有一部機器，當指定所有高速公路的交通情況後，它就能計算出穿梭二個城市 S 和 T 之間所需要的最少總收費 ($S \neq T$)。

然而，這台機器只是一個原型。所以 S 和 T 的值只能是固定的（即它已經在內建在機器的更件內），然而你並不知道它們的值是什麼。你的任務就是去找出 S 和 T 的值。為了計算出答案，你計劃先在機器內指定每條高速公路的交通情況，然後就可以利用它輸出的收費來推斷出 S 和 T 的值。因為指定每條高速公路的交通情況代價不小，所以你並不想使用這台機器太多次。

實現細節

你需要實現下面的程序：

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

- N: 城市的數目。
- U 及 V: 長度為 M 的陣列，其中 M 為連接城市的高速公路的數目。 對於每個 i ($0 \leq i \leq M-1$)， 高速公路 i 連接城市 `U[i]` 和 `V[i]`。
- A: 交通情況順暢時高速公路的收費。
- B: 交通情況繁忙時高速公路的收費。
- 在每個測試樣例中，該函數只會被調用一次。
- 請注意 M 的值為陣列的長度，其實際數值可以通過實現注意事項所提及的方法取得。

程序 `find_pair` 可以調用以下函數:

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

- w 的長度必為 M。 陣列 w 描述高速公路的交通情況。
- 對於每個 i ($0 \leq i \leq M-1$)， `w[i]` 描述高速公路 i 的交通情況。 `w[i]` 的值一定為 0 或 1。

    - `w[i] = 0` 表示高速公路 i 交通情況順暢。
    - `w[i] = 1` 表示高速公路 i 交通情況繁忙。
- 該函數的返回值是在 `w` 描述的交通狀況下，穿梭城市 S 和 T 之間所需的最少總收費。
- 該函數最多只能被調用 100 次（對於每個測試樣例）。

`find_pair` 應調用以下程序以獲取答案：

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

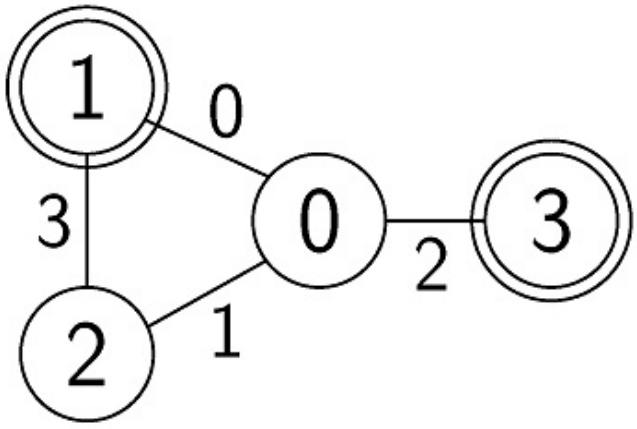
- `s` 和 `t` 的值必需為城市 S 和 T（兩者的先後次序並不重要）。
- 該程序一定會被調用且只會被調用一次。

如果不滿足上面的條件，你的程式將被判視 **`Wrong Answer`**。否則，你的程式將被判為 **`Accepted`**，而你的得分將根據 `ask` 的調用次數來計算（參見子任務）。

樣例

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

評分程式調用 `find_pair(4, [0, 0, 0, 1], [1, 2, 3, 2], 1, 3)`.



在上圖中，每條邊上面的數字 i 是代表該邊是對應公路 i 的。其中一些可能的詢問調用 `ask` 和它相關的返回值如下表所示:

調用	返回值
`ask([0, 0, 0, 0])`	2
`ask([0, 1, 1, 0])`	4
`ask([1, 0, 1, 0])`	5
`ask([1, 1, 1, 1])`	6

對於函數調用 `ask([0, 0, 0, 0])`，所有高速公路的交通情況都是順暢，因此費用都是 1。從城市 $S = 1$ 到城市 $T = 3$ 最便宜的路徑就是 $1 \to 0 \to 3$。這條路徑的總費用等於 2。因此，這個函數的返回值就是 2。

對於一個正確解答，子程序 `find_pair` 應該調用 `answer(1, 3)` 或 `answer(3, 1)`.

附件壓縮包中的檔 `sample-01-in.txt` 就是對應於本樣例的。 其他範例的輸入也在這個壓縮包中。

限制條件

- $2 \le N \le 90\,000$
- $1 \le M \le 130\,000$
- $1 \le A < B \le 1\,000\,000\,000$
- 對於每個 $0 \le i \le M - 1$
  - $0 \le U[i] \le N - 1$
  - $0 \le V[i] \le N - 1$
  - $U[i] \ne V[i]$
- $(U[i], V[i]) \ne (U[j], V[j])$ and $(U[i], V[i]) \ne (V[j], U[j])$ $(0 \le i < j \le M - 1)$
- 從任何一個城市出發你都能夠到達另外任何一個城市。
- $0 \le S \le N - 1$
- $0 \le T \le N - 1$
- $S \ne T$

在本題中，評分程式不是適應性的。意思是說，在評分程式開始運行的時候 S and T 就固定下來，而且不依賴於你的程式所做的詢問。

子任務

1. (5 points) 其中一個 S 或 T 是 $0, N \le 100, M = N - 1$
2. (7 points) 其中一個 S 或 T 是 $0, M = N - 1$
3. (6 points) $M = N - 1, U[i] = i, V[i] = i + 1$ $(0 \le i \le M - 1)$
4. (33 points) $M = N - 1$
5. (18 points) $A = 1, B = 2$
6. (31 points) 沒有附加限制。

如果你的程式被判為**Accepted**，你調用了X次函數`ask`。你獲得的分數P取決於你在那一個子任務，它的計算公式如下：

- 子任務 1。 $P = 5$.
- 子任務 2。 如果 $X \le 60, P = 7$。 否則 $P = 0$.
- 子任務 3。 如果 $X \le 60, P = 6$。 否則 $P = 0$.
- 子任務 4。 如果 $X \le 60, P = 33$。 否則 $P = 0$.
- 子任務 5。 如果 $X \le 52, P = 18$。 否則 $P = 0$.
- 子任務 6。

- 如果 $X \leq 50, P = 31$。
- 如果 $51 \leq X \leq 52, P = 21$。
- 如果 $53 \leq X, P = 0$。

注意，你在每個子任務上的得分，等於你在該子任務所有測試數據中的最低得分。

樣例評分程式

樣例評分程式將讀取如下格式的輸入：

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

如果你的程式被判為**Accepted**，樣例評分程式將列印出 `Accepted: q`，這裏的 `q` 為函數 `ask`被調用的次數。

如果你的程式被判為**Wrong Answer**，它列印出`Wrong Answer: MSG`。而 `MSG` 會是以下其中之一：

- `answered not exactly once`: 函數 `answer` 不是只被調用一次。
- `w is invalid`：提供給函數 `ask` 的 `w` 的長度不是 M 或在對於某些 i $(0 \leq i \leq M - 1)$, `w[i]` 的值既不是 0 也不是 1 .
- `more than 100 calls to ask`: 函數 `ask` 的調用次數超過 100 次。
- `{s, t} is wrong`: 函數 `answer` 被調用時使用了錯誤的 `s` 和 `t`.