Otoyol Ücretleri

Japonya'da, şehirler birbirlerine bir otoyol ağı ile bağlanmışlardır. Bu ağda N şehir ve M otoyol vardır. Her bir otoyol birbirlerinden farklı iki farklı şehri birbirine bağlar. Herhangi farklı iki otoyol aynı şehir çiftini birbirine bağlamaz. Şehirler 0'dan $N-1$'e ve otoyollar 0'dan $M-1$'e numaralandırılmışlardır. Herhangi bir otoyolda her iki yönde de gidilebilir. Otoyolları kullanarak herhangi bir şehirden başka herhangi bir şehre gidebilirsiniz.

Her bir otoyol ücretlendirilmiştir. Bir otoyolun ücreti o otoyoldaki o anki **trafik** yoğunluğuna bağlıdır. Trafik ya **seyrek** ya da **yoğun**dur. Trafik seyrek olduğu zaman, ücret A yendir (Japon parasıyla). Trafik yoğun olduğu zaman, ücret B yendir. $A < B$ olduğu garanti edilmiştir. Ayrıca A ve B değerlerinin ne oldukları da sizlere verilmiştir.

Bütün otoyollardaki o anki trafik yoğunlukları verildiğinde, S ve T ($S \neq T$) şehirleri arasında gitmek için ödemeniz gereken en az ücreti hesaplayan bir makineniz vardır (verilen trafik koşullarına göre).

Fakat bu makine sadece bir prototiptir. S ve T değerleri belirli iki şehre sabitlenmiştir (i.e., makine içine sabit olarak elle yazılmıştır) ve bu şehirlerin ne oldukları size söylenmemiştir. Hedefiniz S ve T değerlerini bulmaktır. Bu hedef doğrultusunda, makineye birçok farklı trafik durumları sunup, makinenin söylediği ücretlere göre S ve T'nin ne olduğunu tahmin etmeyi planladınız. Farklı trafik durumları belirtmek pahalı bir işlem olduğu için makineyi çok kez kullanmak istemiyorsunuz.

Kodlama Detayları

Aşağıdaki prosedürü yazmalısınız:

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

- `N`: şehir sayısı.
- `U` ve `V`: M uzunluğunda diziler. M şehirleri birbirine bağlayan otoyol sayısıdır. Her bir i ($0 \leq i \leq M-1$) için, Otoyol i, `U[i]` ve `V[i]` şehirlerini birbirine bağlar.
- `A`: trafik seyrek olduğunda ödenmesi gereken ücret.
- `B`: trafik yoğun olduğunda ödenmesi gereken ücret.
- Her bir test case için bu prosedür tam olarak bir kez çağrılır.
- M dizilerin uzunluğudur ve Programlama Duyuruları dokümanında belirtildiği gibi elde edilebilir.

`find_pair` prosedürü aşağıdaki fonksiyonu çağırabilir.

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

- `w`'nin uzunluğu M olmalıdır. `w` dizisi trafik yoğunluk durumlarını belirtir.
- Her bir i ($0 \leq i \leq M-1$) için, `w[i]`, i. otoyoldaki trafik yoğunluk durumunu gösterir. `w[i]`'nin değeri ya 0 ya da 1 olmalıdır.
  - `w[i] = 0` i. otoyolda trafik seyrek demektir.
  - `w[i] = 1` i. otoyolda trafik yoğun demektir.
- Bu fonksiyon, `w` tarafından belirtilen trafik yoğunlukları gözönüne alındığında S ve T ($S \neq T$) şehirleri arasında gitmek için ödemeniz gereken en az ücreti döner.
- Bu fonksiyon en fazla 100 kez çağrılabilir (her bir test case için).

`find_pair` fonksiyonunda, cevabı belirtmek için aşağıdaki prosedürü çağırmalısınız:

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

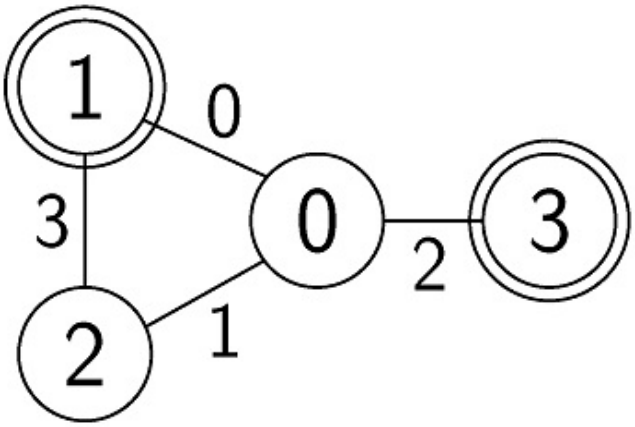
- `s` ve `t` değerleri S ve T ikilisini belirtmelidir (sıralama önemli değildir).
- Bu prosedür tam olarak bir kez çağrılmalıdır.

Yukarıdaki koşullardan bazıları sağlanmazsa, programınız **`Wrong Answer`** (Yanlış Cevap) olarak değerlendirilir. Koşullar sağlandığı takdirde, programınız **`Accepted`** (Kabul Edildi) olarak değerlendirilir ve puanınız `ask` prosedürünü kaç kez çağırdığınıza göre belirlenir (Altgörevlere bakınız).

Örnek

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

Değerlendirici `find_pair(4, [0, 0, 0, 1], [1, 2, 3, 2], 1, 3)` fonksiyonunu çağırır.



Yukarıdaki şekilde, i ile numaralandırılmış kenar, i no'lu otoyola karşılık gelmektedir.

`ask` fonksiyonuna yapılan bazı olası çağrılar ve dönen değerler aşağıda listelenmiştir:

Çağrı	Dönen değer
`ask([0, 0, 0, 0])`	2
`ask([0, 1, 1, 0])`	4
`ask([1, 0, 1, 0])`	5
`ask([1, 1, 1, 1])`	6

`ask([0, 0, 0, 0])` çağrısı için, bütün otoyollarda trafik seyrektir ve her bir otoyol ücreti 1'dir. $S = 1$'den $T = 3$'e en ucuz rota $1 \to 0 \to 3$ rotasıdır. Bu rota için toplam ücret 2'dir. O nedenle bu fonksiyon 2 döner.

Doğru cevabı belirtmek için `find_pair` prosedürü `answer(1, 3)`'i ya da `answer(3, 1)`'i çağırmalıdır.

Ekteki ziplenmiş paketteki `sample-01-in.txt` dosyası bu örneğe karşılık gelmektedir. Pakette bundan başka örnek girdiler de bulunmaktadır.

Kısıtlar

- $2 \leq N \leq 90\,000$
- $1 \leq M \leq 130\,000$
- $1 \leq A < B \leq 1\,000\,000\,000$
- Her bir $0 \leq i \leq M - 1$ için
  - $0 \leq U[i] \leq N - 1$
  - $0 \leq V[i] \leq N - 1$
  - $U[i] \neq V[i]$
- $(U[i], V[i]) \neq (U[j], V[j])$ ve $(U[i], V[i]) \neq (V[j], U[j])$ $(0 \leq i < j \leq M - 1)$
- Otoyolları kullanarak herhangi bir şehirden başka herhangi bir şehre gidebilirsiniz.
- $0 \leq S \leq N - 1$
- $0 \leq T \leq N - 1$
- $S \neq T$

Bu soruda, değerlendirici kendini dinamik olarak uyarlamaz. Bunun anlamı S ve T 'nin değerlendiricinin çalışmasının başında sabitlenmiş olmaları ve çözümünüz tarafından yapılan sorgulara bağlı olarak değişmemeleridir.

Altgörevler

1. (5 puan) S ya da T'den birisi 0, $N \leq 100$, $M = N - 1$
2. (7 puan) S ya da T'den birisi 0, $M = N - 1$

3. (6 puan) $M = N - 1$, $U[i] = i$, $V[i] = i + 1$ $(0 \leq i \leq M - 1)$
4. (33 puan) $M = N - 1$
5. (18 puan) $A = 1$, $B = 2$
6. (31 puan) Ek kısıt bulunmamaktadır

Programınızın **`Accepted`** (Kabul edildi) olarak değerlendirilmiş olduğunu ve X tane `ask` çağrısı yapmış olduğunuzu varsayalım. Bu durumda bu test case için puanınız P, altgörev numarasına bağlı olarak, aşağıdaki gibi hesaplanır:

- Altgörev 1. $P = 5$.
- Altgörev 2. Eğer $X \leq 60$ ise $P = 7$ Değilse $P = 0$.
- Altgörev 3. Eğer $X \leq 60$ ise, $P = 6$ Değilse $P = 0$.
- Altgörev 4. Eğer $X \leq 60$ ise, $P = 33$ Değilse $P = 0$.
- Altgörev 5. Eğer $X \leq 52$ ise, $P = 18$ Değilse $P = 0$.
- Altgörev 6.
  - Eğer $X \leq 50$ ise, $P = 31$.
  - Eğer $51 \leq X \leq 52$ ise, $P = 21$.
  - Eğer $53 \leq X$ ise, $P = 0$.

Her bir altgörev için puanınız o altgörevdeki bütün test case'lerden alınan puanların minimumuna eşittir.

Örnek değerlendirici

Örnek değerlendirici girdiyi aşağıdaki formatta okur:

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

Programınız **`Accepted`** (Kabul Edildi) olarak değerlendirilirse, örnek değerlendirici `Accepted: q` basar, `q` `ask` fonksiyonuna yapılan çağrıların sayısını gösterir.

Programınız **`Wrong Answer`** (Yanlış Cevap) olarak değerlendirilirse, örnek değerlendirici `Wrong Answer: MSG` basar, ve `MSG` aşağıdakilerden birisi olur:

- `answered not exactly once`: `answer` prosedürü tam olarak bir kez çağrılmamıştır.
- `w is invalid`: `ask`'e gönderilen `w`'nin uzunluğu M değildir ya da `w[i]` bazı i ($0 \leq i \leq M - 1$) değerleri için 0 ya da 1 değildir.
- `more than 100 calls to ask`: `ask` fonksiyonu 100'den fazla çağrılmıştır.
- `{s, t} is wrong`: `answer` prosedürü yanlış `s` ve `t` ikilisi ile çağrılmıştır.