Simurq

"Şahnamə"də nəql olunan Qədim Fars mifologiyasına əsasən, əfsanəvi qəhrəman Zal, Kabil şəhzadəsi Rüdabəyə dəlicəsinə aşiqdir. Zal onunla evlənmək istədikdə, Rüdabənin atası Zalı sınağa çəkmək qərarına gəlir.

İranda 0-dan $n-1$-ə qədər nömrələnmiş n sayda şəhər və 0-dan $m-1$-ə qədər nömrələnmiş m sayda ikitərəfli yol var. Hər bir yol iki müxtəlif şəhəri birləşdirir. İstənilən iki şəhər ən çoxu bir yol vasitəsi ilə birləşir. Bəzi yollar *şah yollarıdır*, və yalnız şah ailəsi tərəfindən istifadə olunduğuna görə gizli saxlanılır. Zalın tapşırığı *şah yollarını* tapmaqdır.

Zalda İranın bütün şəhərləri və yolları olan xəritə var. O, bu yollardan hansılarının *şah yolları* olduğunu bilmir, amma Zalın qoruyucusu olan xeyirxah əfsanəvi Simurq quşu ona kömək edə bilər. Ancaq Simurq şah yollarını birbaşa açıqlamaq istəmir. Bunun əvəzinə, o, Zala şah yolları çoxluğunun *qızıl çoxluq* olduğunu deyir. Yollar çoxluğu yalnız və yalnız o zaman qızıl çoxluq olur ki,

- onun tərkibində $n-1$ sayda yol olsun və
- yalnız bu çoxluğun yollarından istifadə etməklə istənilən şəhərdən digərinə getmək olar.

Bundan əlavə, Zal Simurqa bir neçə sual verə bilər. Hər sual üçün:

1. Zal yolların *qızıl çoxluğunu* seçir, və daha sonra
2. Simurq Zala seçilmiş qızıl çoxluqda olan yolların neçəsinin şah yolu olduğunu söyləyir.

Sizin proqramınız Simurqa ən çoxu q sual verməklə, Zalın şah yolları çoxluğunu tapmasına yardım etməlidir. Yoxlayıcı sistem Simurq rolunu oynayır.

Gerçəkləşdirmə detalları

Aşağıdakı proseduru gerçəkləşdirməlisiniz:

```
int[] find_roads(int n, int[] u, int[] v)
```

- n: şəhərlərin sayıdır,
- u və v: m ölçülü massivlərdir. Hər bir $0 \leq i \leq m-1$ üçün, i yolu vasitəsi ilə $u[i]$ və $v[i]$ şəhərləri birləşir.
- Bu prosedur ölçüsü $n-1$ olan, şah yollarının nömrələrini saxlayan (istənilən ardıcıllıqla) massiv qaytarmalıdır.

Sizin həlliniz aşağıdakı yoxlayıcı sistem prosedurunu ən çoxu q dəfə çağıra bilər:

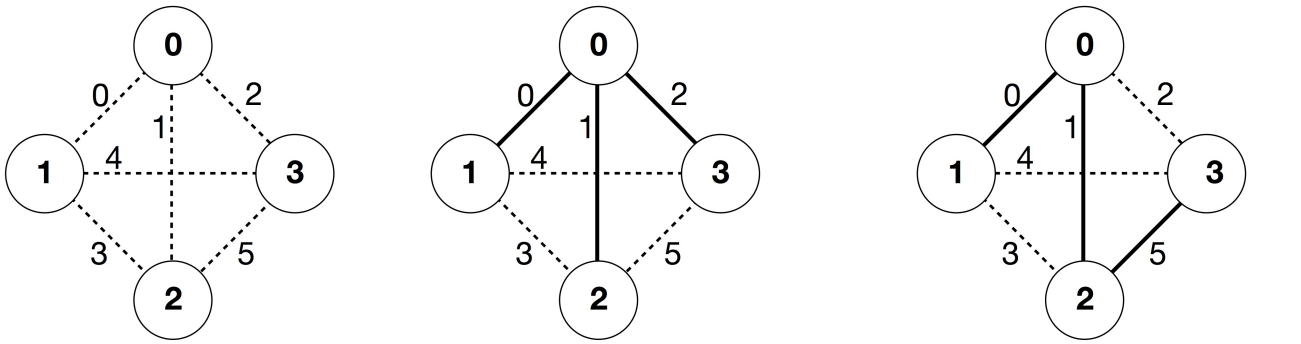
```
int count_common_roads(int[] r)
```

- r: ölçüsü $n-1$ olan, qızıl çoxluğun yollarının nömrələrini saxlayan (istənilən ardıcıllıqla) massivdir.
- Bu prosedur r massivində olan şah yollarının sayını qaytarır.

Nümunə

```
find_roads(4, [0, 0, 0, 1, 1, 2], [1, 2, 3, 2, 3, 3])
```

`find_roads(...)` `count_common_roads([0, 1, 2])` = 2 `count_common_roads([5, 1, 0])` = 3



Bu nümunədə 4 şəhər və 6 yol var. (a, b) vasitəsilə a və b şəhərlərini birləşdirən yol göstərilir. Yollar 0-dan 5-ə qədər növbəti ardıcıllıqda nömrələnmişdir: $(0, 1)$, $(0, 2)$, $(0, 3)$, $(1, 2)$, $(1, 3)$, and $(2, 3)$. Hər bir qızıl çoxluq $n - 1 = 3$ yoldan ibarətdir.

Tutaq ki, 0, 1 və 5 nömrəli yollar, yəni $(0, 1)$, $(0, 2)$, və $(2, 3)$, şah yollarıdır. Onda:

- `count_common_roads([0, 1, 2])` nəticəsi 2 olacaq. Bu sorğu $0, 1,$ və 2 nömrəli yollar, yəni $(0, 1)$, $(0, 2)$ və $(0, 3)$ yolları barədədir. Bunlardan ikisi şah yoludur.
- `count_common_roads([5, 1, 0])` nəticəsi 3 olacaq. Bu sorğu bütün şah yolları barədədir.

`find_roads` proseduru ya `[5, 1, 0]`, və yaxud ölçüsü 3 olan və bu elementləri özündə saxlayan istənilən bir massiv qaytarmalıdır.

Qeyd edək ki, aşağıdakı sorğular düzgün deyil:

- `count_common_roads([0, 1])`: burada r-in ölçüsü 3 deyil.
- `count_common_roads([0, 1, 3])`: burada r qızıl çoxluq deyil, çünki 0 nömrəli şəhərdən 3 nömrəli şəhərə yalnız $(0, 1)$, $(0, 2)$, $(1, 2)$ yollarından istifadə etməklə getmək mümkün deyil.

Məhdudiyyətlər

- $2 \leq n \leq 500$
- $n-1 \leq m \leq n(n-1)/2$
- $0 \leq u[i], v[i] \leq n-1$ (hər bir $0 \leq i \leq m-1$ üçün)
- Hər bir $0 \leq i \leq m-1$ üçün, i yolu iki müxtəlif şəhəri birləşdirir (i.e., $u[i] \neq v[i]$).
- İstənilən iki şəhər arasında ən çoxu bir yol var.
- Yollar vasitəsi ilə istənilən iki şəhər arasında səyahət etmək mümkündür.
- Bütün şah yolları çoxluğu qızıl çoxluqdur.
- `find_roads` proseduru `count_common_roads` prosedurunu ən çoxu q çağırmalıdır. Hər sorğuda r çoxluğunda təyin edilmiş yollar qızıl çoxluq təşkil etməlidir.

Altməsələlər

1. (13 xal) $n \leq 7$, $q = 30\,000$
2. (17 xal) $n \leq 50$, $q = 30\,000$
3. (21 xal) $n \leq 240$, $q = 30\,000$
4. (19 xal) $q = 12000$ və istənilən iki şəhər arasında bir yol var.
5. (30 xal) 8000

Nümunə yoxlayıcı

Nümunə yoxlayıcı giriş verilənlərini aşağıdakı formatda oxuyur:

- sətir 1: n m
- sətir $2 + i$ (for all $0 \leq i \leq m-1$): $u[i]$ $v[i]$
- sətir $2 + m$: $s[0]$ $s[1]$ $\ldots$ $s[n-2]$

Burada $s[0], s[1], \ldots, s[n-2]$ şah yollarının nömrələridir.

Nümunə yoxlayıcı yalnız o zaman `YES` cavab verir ki, `find_roads` proseduru `count_common_roads` prosedurunu ən çoxu $30\,000$ dəfə çağırmış olsun, və nəticə olaraq düzgün şah yolları çoxluğunu qaytarsın. Əks halda cavab `NO` olacaqdır.

Diqqət edin ki, nümunə yoxlayıcı olan `count_common_roads` proseduru r çoxluğunun qızıl çoxluğa məxsus bütün xüsusiyyətlərə malik olmasını yoxlamır. Bunun yerinə, o, sadəcə r massivində olan şah yollarını sayır və nəticə olaraq qaytarır. Buna baxmayaraq, əgər sizin proqramınız `count_common_roads` prosedurunu qızıl çoxluq təşkil etməyən nömrələrlə çağırarsa, yoxlamanın nəticəsi 'Wrong Answer' olacaqdır.

Texniki qeyd

`count_common_roads` proseduru, effektivlik səbəbinə görə, C++ və Pascal dillərində massivin özünü deyil, onun ünvanını (*pass by reference*) qəbul edir. Bununla belə, proseduru həmişəki kimi adi qaydada çağıra bilərsiniz. Yoxlayıcı sistem r massivində olan qiymətlərin dəyişməyəcəyinə təminat verir.