Սիմուրղ

Համաձայն Շահնամեի հին պարսկական լեգենդների, պարսիկ լեգենդար հերոս Զալը սիրահարված էր Քաբուլի արքայադուստր Ռուդեբային: Երբ Զալը խնդրեց Ռուդեբայի ձեռքը, աղջկա հայրը Զալին խնդիր առաջադրեց:

Պարսկաստանում կա n հատ քաղաք, որոնք համարակալված 0-ից $n-1$ և m հատ երկկողմանի ճանապարհ, որոնք համարակալված 0-ից $m-1$: Յուրաքանչյուր ճանապարհ միացնում է երկու տարբեր քաղաքներ: Յուրաքանչյուր երկու քաղաք միացված են առավելագույնը մեկ ճանապարհով: Որոշ ճանապարհներ *թագավորական* են, որոնցով միայն թագավորական շքախմբերն են անցնում: Զալի խնդիրն է պարզել, թե որ ճանապարհներն են թագավորական:

Զալն ունի Պարսկաստանի բոլոր քաղաքների և ճանապարհների քարտեզ: Նա չգիտի, թե որ ճանապարհներն են թագավորական, բայց նա կարող է օգնություն ստանալ բարեսիրտ հեքիաթային թռչուն Սիմուրղից, որը Զալի պահապանն է: Սակայն Սիմուրղը չի ցանկանում ուղղակիորեն բացահայտել թագավորական ճանապարհների բազմությունը: Փոխարենը նա Զալին ասում է, որ թագավորական ճանապարհների բազմությունը *ոսկե բազմություն* է: Ճանապարհների բազմությունը ոսկե բազմություն է այն և միայն այն դեպքում,

- եթե նրանում կա *ճիշտ* $n-1$ ճանապարհ, և
- միայն այդ ճանապարհներով հնարավոր է երկրի ցանկացած քաղաքից հասնել մեկ այլ ցանկացած քաղաք:

Ավելին, Զալը կարող է Սիմուրղին որոշակի քանակությամբ հարցեր տալ: Յուրաքանչյուր հարցի համար

1. Զալն ընտրում է ճանապարհների *ոսկե* բազմություն, ապա
2. Սիմուրղն ասում է Զալին, թե ընտրված ոսկե բազմության ճանապարհներից քանիսն են թագավորական:

Ձեր ծրագիրը պետք է օգնի Զալին պարզելու թագավորական ճանապարհների բազմությունը Սիմուրղին տալով առավելագույնը q հարց: Գրեյդերը կատարելու է Սիմուրղի դերը:

Իրականացման մանրամասներ

Դուք պետք է իրականացնեք հետևյալ ֆունկցիան.

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

- n-ը քաղաքների քանակն է,
- u-ն և v-ն m երկարությամբ զանգվածներ են: $u[i]$-ն և $v[i]$-ն i-րդ ճանապարհով միացված քաղաքներն են, $0 \leq i \leq m-1$:
- Այս ֆունկցիան պետք է վերադարձնի թագավորական ճանապարհների համարները (կամայական կարգով) պարունակող $n-1$ երկարությամբ զանգված:

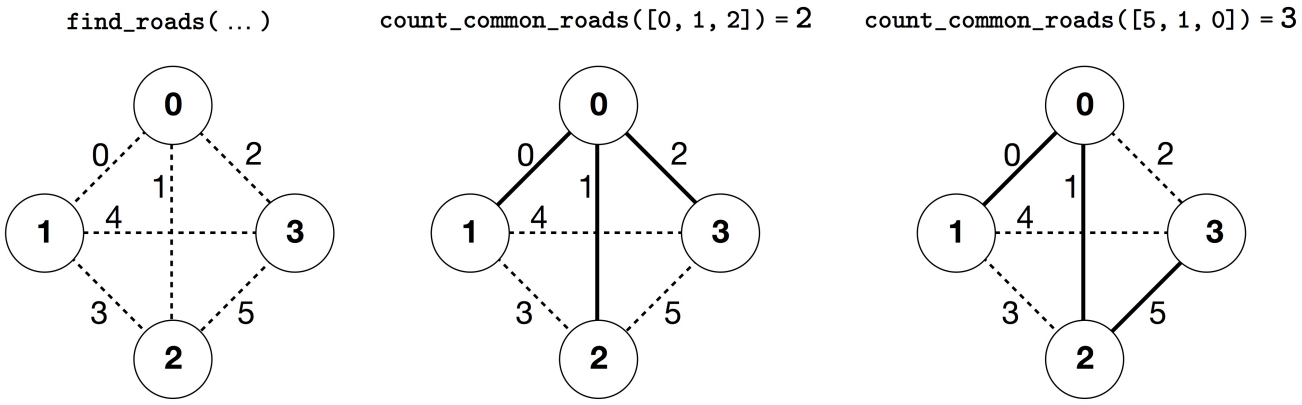
Ձեր լուծումը կարող է կատարել գրեյդերի հետևյալ ֆունկցիայի առավելագույնը q կանչ.

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

- r-ը ոսկե բազմությանը պատկանող ճանապարհների $n-1$ երկարությամբ զանգված է (տրված կամայական կարգով):
- Այս ֆունկցիան վերադարձնում է r-ում թագավորական ճանապարհների քանակը:

Օրինակ

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



Այս օրինակում կա 4 քաղաք և 6 ճանապարհ: (a, b)-ով նշանակենք a և b քաղաքները միացնող ճանապարհը: Ճանապարհները համարակալված են 0-ից մինչև 5 հետևյալ կարգով՝ $(0, 1)$, $(0, 2)$, $(0, 3)$, $(1, 2)$, $(1, 3)$ և $(2, 3)$: Յուրաքանչյուր ոսկե բազմություն ունի $n-1=3$ ճանապարհ:

Ենթադրենք, որ թագավորական ճանապարհների համարներն են 0-ն, 1-ը և 5-ը, այսինքն, $(0, 1)$, $(0, 2)$ և $(2, 3)$ ճանապարհները: Հետևաբար,

- `count_common_roads([0, 1, 2])`-ը վերադարձնում է 2: Այս հարցումը վերաբերում է $0, 1$ և 2 համարներով ճանապարհներին, այսինքն $(0, 1)$, $(0, 2)$ և

$(0, 3)$ ճանապարհներին: Նրանցից երկուսը թագավորական են:
- `count_common_roads([5, 1, 0])`-ը վերադարձնում է 3: Այս հարցումը վերաբերում է թագավորական ճանապարհների բազմությանը:

`find_roads` ֆունկցիան պետք է վերադարձնի `[5, 1, 0]` կամ այդ երեք տարրերը պարունակող 3 երկարությամբ ցանկացած այլ զանգված:

Նկատենք, որ հետևյալ կանչերը անթույլատրելի են.

- `count_common_roads([0, 1])`. այստեղ r-ի երկարությունը 3 չէ:
- `count_common_roads([0, 1, 3])`. այստեղ r-ի նկարագրածը ոսկե բազմություն չէ, որովհետև հնարավոր չէ 0 քաղաքից հասնել 3 քաղաքը, օգտագործելով միայն $(0, 1)$, $(0, 2)$, $(1, 2)$ ճանապարհները:

Սահմանափակումներ

- $2 \le n \le 500$
- $n - 1 \le m \le n(n-1)/2$
- $0 \le u[i], v[i] \le n - 1$ (for all $0 \le i \le m - 1$)
- i-րդ ճանապարհը, $0 \le i \le m - 1$, միացնում է երկու տարբեր քաղաքներ (այսինքն, $u[i] \neq v[i]$):
- Յուրաքանչյուր երկու քաղաքներ միացված են առավելագույնը մեկ ճանապարհով:
- Տրված ճանապարհներով ցանկացած քաղաքից կարելի է հասնել ցանկացած մեկ այլ քաղաք:
- Բոլոր թագավորական ճանապարհների բազմությունը ոսկե բազմություն է:
- `find_roads`-ում `count_common_roads`-ը պետք է կանչվի առավելագույնը q անգամ: Յուրաքանչյուր կանչում r-ով նկարագրված ճանապարհների բազմությունը պետք է լինի ոսկե բազմություն:

Ենթախնդիրներ

1. (13 միավոր) $n \le 7$, $q = 30\,000$
2. (17 միավոր) $n \le 50$, $q = 30\,000$
3. (21 միավոր) $n \le 240$, $q = 30\,000$
4. (19 միավոր) $q = 12\,000$ յուրաքանչյուր երկու քաղաք միացված են ճանապարհով
5. (30 միավոր) $q = 8000$

Գրեյդերի օրինակ

Գրեյդերի օրինակը կարդում է մուտքային տվյալները հետևյալ ձևաչափով.

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

Այստեղ $s[0]$-ն, $s[1]$-ը, $\ldots$, $s[n-2]$-ը թագավորական ճանապարհների համարներն են:

Գրեյդերի օրինակը արտածում է `YES`, եթե `find_roads`-ում `count_common_roads`-ը կանչվել է առավելագույնը $30\,000$ անգամ և վերադարձրել է թագավորական ճանապարհների ճիշտ բազմությունը: Հակառակ դեպքում այն արտածում է `NO`:

Զգույշ եղեք. գրեյդերի օրինակում `count_common_roads` ֆունկցիան չի ստուգում r-ը ունի ոսկե բազմության բոլոր հատկությունները, թե ոչ: Փոխարենը, հաշվում և վերադարձնում է r զանգվածում թագավորական ճանապարհների քանակը: Սակայն, եթե ձեր ուղարկած ծրագրում `count_common_roads` ֆունկցիան կանչվի համարների այնպիսի բազմությամբ, որը ոսկե չէ, կստանաք 'Wrong Answer':

Տեխնիկական դիտողություն

`count_common_roads` ֆունկցիան C++-ում և Pascal-ում օգտագործում է *call by reference* եղանակը արդյունավետության նկատառումներով: Դուք կարող եք ֆունկցիան կանչել սովորական եղանակով: Երաշխավորվում է, որ գրեյդերը r-ի արժեքները չի փոխում: