Simurgh

Nelle antiche leggende Persiane di Shahnameh, il leggendario eroe Zal è perdutamente innamorato di Rudaba, principessa di Kabul. Quando Zal chiede la mano di Rudaba, il padre gli propone una sfida per dimostrare il suo valore.

In Persia ci sono n città (numerate da 0 a $n-1$) ed m strade bidirezionali (numerate da 0 a $m-1$), che Zal conosce alla perfezione. Ogni strada connette una coppia di città distinte, e ogni coppia di città è connessa da al massimo una strada.

Alcune delle strade sono *strade regali*, ma nessuno sa quali sono: come sfida, Zal deve trovarle tutte. Per farlo può ottenere indizi da Simurgh (il suo mitico uccello protettore). Come primo indizio Simurgh gli rivela che l'insieme delle strade regali è *dorato*, vale a dire:

- ha *esattamente* $n-1$ strade,
- per ogni coppia di città è possibile raggiungere una partendo dall'altra, percorrendo solo strade dell'insieme.

Inoltre, Simurgh consente a Zal di fare **al massimo** q tentativi, nei quali:

1. Zal sceglie un insieme *dorato* di strade;
2. Simurgh risponde dicendo quante delle strade proposte da Zal sono effettivamente *regali*.

Aiuta Zal a trovare le strade regali facendo al massimo q tentativi con Simurgh.

Dettagli di implementazione

Devi implementare la seguente funzione:

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

- n: il numero di città,
- u, v: due array di lunghezza m tali che le città $u[i]$ e $v[i]$ sono connesse dalla strada i (per ogni $0 \leq i \leq m-1$).
- La funzione deve restituire un array di lunghezza $n-1$ contenente gli indici corrispondenti a strade regali (in un qualunque ordine).

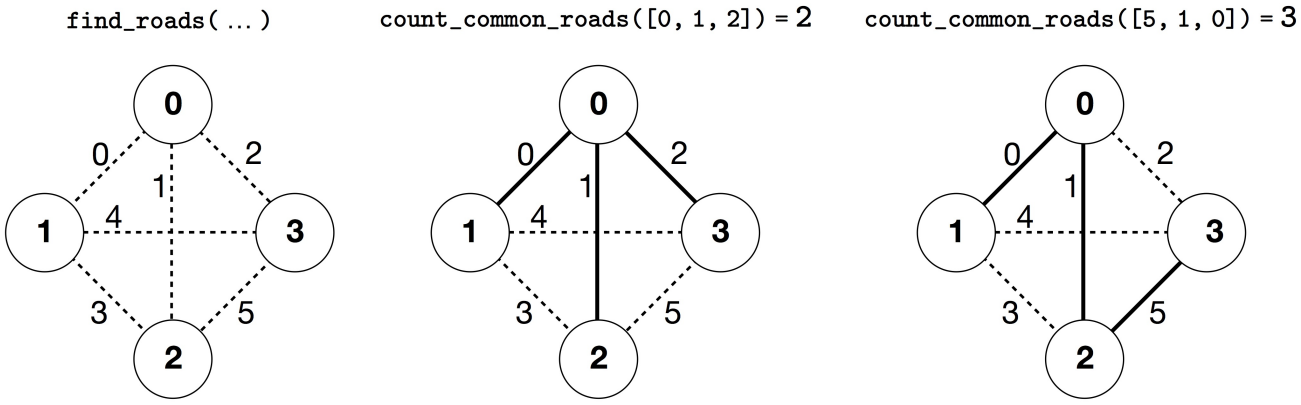
La tua soluzione può fare al massimo q chiamate alla seguente funzione ausiliaria definita nel grader:

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

- r: un array di lunghezza $n-1$ contenente degli indici di strade in un insieme dorato (in un qualunque ordine).
- La funzione restituisce il numero di strade regali in r (per **common** nel nome della funzione intendiamo le strade *in comune* con l'insieme dorato delle **strade regali**).

Esempio

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



In questo esempio ci sono 4 città e 6 strade: denotiamo con (a, b) la strada che connette le città a e b. Le strade sono indicizzate da 0 a 5 nel seguente ordine: $(0, 1)$, $(0, 2)$, $(0, 3)$, $(1, 2)$, $(1, 3)$, e $(2, 3)$. Ogni insieme dorato consiste di $n - 1 = 3$ strade.

Supponiamo che le strade regali siano quelle indicizzate con 0, 1, 5, ovvero le strade $(0, 1)$, $(0, 2)$, $(2, 3)$.

- `count_common_roads([0, 1, 2])` restituisce 2: infatti, questo tentativo riguarda le strade $0, 1, 2$ (vale a dire $(0, 1)$, $(0, 2)$, $(0, 3)$), di cui due sono regali.
- `count_common_roads([5, 1, 0])` restituisce 3: infatti, questo tentativo riguarda l'intero insieme di strade regali.

La funzione `find_roads` deve quindi restituire `[5, 1, 0]` o un qualunque altro array di lunghezza 3 contenente questi stessi elementi in un qualche ordine.

Nota che le seguenti chiamate non sono consentite:

- `count_common_roads([0, 1])`: la lunghezza di r non è 3.
- `count_common_roads([0, 1, 3])`: r non rappresenta un insieme dorato perché non è possibile viaggiare dalla città 0 alla città 3 solamente lungo le strade $(0, 1)$, $(0, 2)$, $(1, 2)$.

Assunzioni

- $2 \leq n \leq 500$.
- $n - 1 \leq m \leq n(n-1)/2$.
- $0 \leq u[i], v[i] \leq n - 1$ (per ogni $0 \leq i \leq m - 1$).
- $u[i] \neq v[i]$, cioè la strada i collega due città diverse (per ogni $0 \leq i \leq m - 1$).
- C'è al massimo una strada che collega ogni coppia di città (non esistono strade duplicate).
- È possibile viaggiare tra ogni coppia di città tramite le strade esistenti.
- L'insieme delle strade regali è dorato.
- `find_roads` deve chiamare `count_common_roads` **al massimo** q volte, e ogni volta l'insieme di strade r specificato deve essere dorato.

Assegnazione del punteggio

1. **(13 punti)** $n \leq 7$, $q = 30\,000$
2. **(17 punti)** $n \leq 50$, $q = 30\,000$
3. **(21 punti)** $n \leq 240$, $q = 30\,000$
4. **(19 punti)** $q = 12000$ e per ogni coppia di città c'è una strada che le collega
5. **(30 punti)** $q = 8000$

Grader di prova

Il grader di prova legge l'input nel seguente formato:

- riga 1: n m
- riga $2 + i$ ($0 \leq i \leq m - 1$): $u[i]$ $v[i]$
- riga $2 + m$: $s[0]$ $s[1]$ $\ldots$ $s[n-2]$ (gli indici delle strade regali)

Il grader di prova stampa `YES` se `find_roads` chiama `count_common_roads` al massimo $30\,000$ volte e restituisce l'insieme corretto di strade regali; altrimenti, stampa `NO`.

Fai attenzione al fatto che la funzione `count_common_roads` nel grader di prova non controlla se r è davvero dorato. Piuttosto, essa conta e restituisce il numero di indici di strade regali nell'array r. Comunque, se il programma che sottoporrai chiamerà `count_common_roads` con un insieme di indici che non descrive un insieme dorato, il verdetto del server di correzione sarà 'Wrong Answer'.

Nota tecnica

La funzione `count_common_roads` in C++ e Pascal usa una *chiamata per riferimento* per motivi di performance. Puoi comunque chiamare la funzione nel modo solito e viene garantito che il grader non cambia mai il valore di r.