Сімург

Відповідно до персидських легенд в Шахнаме, Зал – легендарний персидський герой, який був шалено закоханий в Рудабу, принцесу Кабула. Коли Зал попросив одружитись з ним Рудабу, її батько захотів його випробувати.

В Персії є n міст, пронумерованих від 0 до $n-1$, та m доріг з двостороннім рухом, пронумерованих від 0 до $m-1$. Кожна дорога з'єднує пару різних міст. Кожна пара міст з'єднана не більш ніж однією дорогою. Деякі дороги є *королівськими дорогами*, які використовуються для подорожі членами королівської родини. Завдання Зала визначити, які з доріг є королівськими дорогами.

Зал має карту, на якій позначені усі міста та дороги в Персії. Він не знає, які з доріг є королівськими, але він може отримати допомогу від Сімурга – доброзичливого міфічного птаха, покровителя Зала. Проте, Сімург не хоче одразу розкрити мережу королівських доріг. Замість цього він говорить Залу, що мережа королівських доріг є *золотою мережею доріг*. Мережа доріг є золотою мережею доріг тоді і тільки тоді, коли:

- вона має *рівно* $n-1$ дорогу, та
- для кожної пари міст є можливість потрапити з одного до іншого, подорожуючи дорогами з цієї мережі.

До того ж, Зал може задати Сімургу декілька запитань. Для кожного запитання:

1. Зал обирає *золоту* мережу доріг і потім
2. Сімург розповідає Залу, скільки доріг в обраній золотій мережі є королівськими дорогами.

Ваша програма повинна допомогти Залу знайти мережу королівських доріг, задаючи Сімургу не більше q запитань. Модуль перевірки буде грати роль Сімурга.

Деталі реалізації

Ви повинні реалізувати наступну процедуру:

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

- n: кількість міст,
- u та v: масив довжини m. Для усіх $0 \leq i \leq m-1$, $u[i]$ та $v[i]$ є містами, що зв'язані дорогою i.
- Ця процедура повинна повертати масив довжини $n-1$, який містить номери

королівських доріг (в довільному порядку)

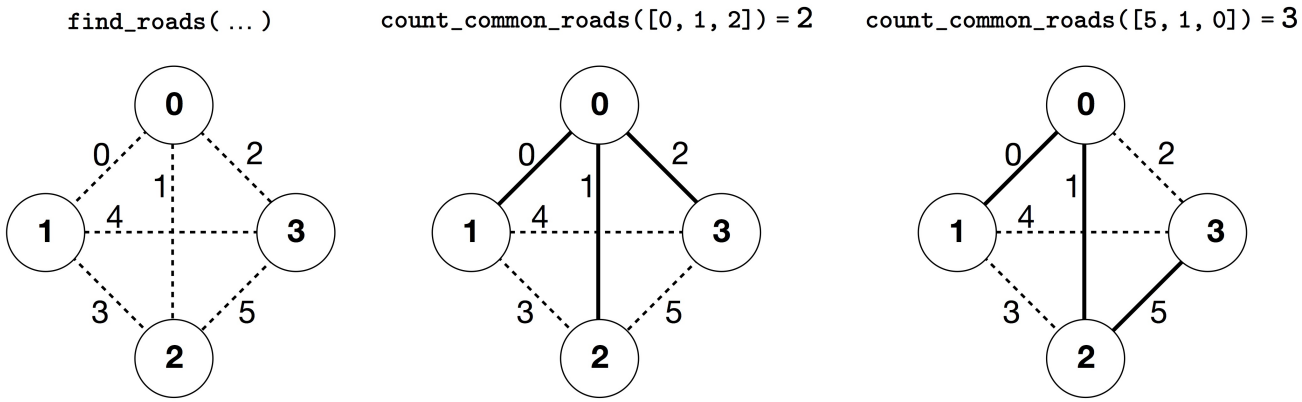
Ваш розв'язок може зробити не більше q викликів наступної процедури з модуля перевірки:

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

- r: масив довжини $n-1$, що містить номери доріг в золотій мережі (в довільному порядку).
- Ця процедура повертає кількість королівських доріг в r.

Приклад

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



В цьому прикладі є 4 міста та 6 доріг. Ми позначаємо як (a, b) дорогу, що з'єднує міста a та b. Дороги пронумеровані від 0 до 5 в наступному порядку: $(0, 1)$, $(0, 2)$, $(0, 3)$, $(1, 2)$, $(1, 3)$, та $(2, 3)$. Кожна золота мережа має $n - 1 = 3$ доріг.

Припустимо, що королівські дороги мають номери 0, 1, та 5, тобто це дороги $(0, 1)$, $(0, 2)$, та $(2, 3)$. Тоді:

- `count_common_roads([0, 1, 2])` повертає 2. Цей запит стосується доріг з номерами $0, 1$, та 2, тобто доріг $(0, 1)$, $(0, 2)$ та $(0, 3)$. Дві з них є королівськими дорогами.
- `count_common_roads([5, 1, 0])` повертає 3. Цей запит стосується мережі усіх королівських доріг.

Процедура `find_roads` повинна повернути `[5, 1, 0]` або будь-який інший масив довжини 3, що містить ці три елементи.

Зверніть увагу, що наступні запити є недопустимими:

- `count_common_roads([0, 1])`: тут довжина r не 3.
- `count_common_roads([0, 1, 3])`: тут r не описує золоту мережу, тому що не можливо потрапити з міста 0 до міста 3 тільки дорогами $(0, 1)$, $(0, 2)$, $(1, 2)$.

Обмеження

- $2 \leq n \leq 500$
- $n - 1 \leq m \leq n(n-1)/2$
- $0 \leq u[i], v[i] \leq n - 1$ (for all $0 \leq i \leq m - 1$)
- Для усіх $0 \leq i \leq m - 1$, дорога i з'єднує два різні міста (тобто, $u[i] \neq v[i]$).
- Є не більше ніж одна дорога між кожною парою міст.
- Можливо подорожувати між будь-якою парою міст дорогами.
- Мережа королівських доріг є золотою мережею.
- `find_roads` повинна викликати `count_common_roads` не більше ніж q разів. В кожному запиті мережа доріг, визначена масивом r, має бути золотою мережею.

Підзадачі

1. (13 балів) $n \leq 7$, $q = 30\,000$
2. (17 балів) $n \leq 50$, $q = 30\,000$
3. (21 бал) $n \leq 240$, $q = 30\,000$
4. (19 балів) $q = 12\,000$ та є дорога між кожною парою міст
5. (30 балів) $q = 8000$

Приклад модуля перевірки

Модуль перевірки з прикладу читає вхідні дані в наступному форматі:

- рядок 1: $\quad n \;\; m$
- рядок $2 + i$ (для всіх $0 \leq i \leq m - 1$): $\quad u[i] \;\; v[i]$
- рядок $2 + m$: $\quad s[0] \;\; s[1] \;\; \ldots \;\; s[n-2]$

Тут $s[0], s[1], \ldots, s[n-2]$ є номерами королівських доріг.

Цей модуль перевірки виводить `YES`, якщо `find_roads` викликає `count_common_roads` не більше $30\,000$ разів і повертає правильну мережу королівських доріг. В іншому випадку він виводить `NO`.

Візьміть до уваги, що процедура `count_common_roads` в прикладі модуля перевірки не перевіряє чи r має властивості золотої мережі. Замість цього вона підраховує та повертає кількість номерів королівських доріг в масиві r. Проте, якщо програма, яку ви відправили викликає `count_common_roads` з набором номерів, що не описують золоту мережу, то відповідь модуля перевірки буде 'Wrong Answer'.

Технічні вимоги

Процедура `count_common_roads` в C++ та Pascal використовує спосіб виклику *за посиланням* з метою ефективності. Ви ж можете робити виклик процедури звичайним шляхом.

Гарантується, що модуль перевірки не змінить значення r.