سميرغ

تقول إحدى الأساطير الفارسية القديمة, بأن الفارس زمشكل وقع في حب أميرة كابل رانيا. وعندما طلب الفارس زمشكل يد الأميرة رانيا للزواج, أعطاه والدها تحدٍ برمجي يجب عليه أن يحله.

لدينا n مدينة في بلاد فارس, مرقمة من 0 إلى $n-1$, بالإضافة إلى m وصلة ثنائية الاتجاه مرقمة من 0 إلى $m-1$. كل وصلة تصل بين مدينتين مختلفتين, وكل مدينتين متصلتين بوصلة واحدة على الأكثر. بعض الوصلات تدعى *وصلات ملكية*, تستخدم للسفر من قبل الملوك. مهمة الفارس زمشكل هي تحديد أي الوصلات هي وصلات ملكية.

يمتلك الفارس زمشكل خريطة بكل المدن وكل الوصلات في بلاد فارس, ولكنه لا يعلم أي من الوصلات هي وصلات ملكية. لمعرفة ذلك يستطيع الاستعانة بالتنين الطائر (سميرغ) الذي يحميه دائماً. هذا التنين لا يرغب بإخباره بالوصلات الملكية بشكل مباشر. بدلاً من ذلك, أخبر التنين الفارس زمشكل بأن جميع الوصلات الملكية تشكل *مجموعة ذهبية*.

تعتبر مجموعة من الوصلات *مجموعة ذهبية* إذا وفقط إذا تحقق:

- تحتوي المجموعة تماماً على $n-1$ وصلة
- ممكن السفر بين أي مدينتين باستخدام وصلات من هذه المجموعة فقط.

يستطيع الفارس زمشكل أن يسأل التنين الطائر مجموعة من الأسئلة, في كل سؤال:

1. يقوم الفارس زمشكل باختيار مجموعة ذهبية من الوصلات
2. ثم يقوم التنين الطائر بإخبار الفارس زمشكل بعدد الطرق الملكية الموجودة ضمن هذه المجموعة الذهبية.

يجب على برنامجك مساعدة الفارس زمشكل على إيجاد مجموعة الوصلات الملكية من خلال سؤال التنين الطائر عل الأكثر q سؤال.

تفاصيل التنجيز

يتوجب عليك تنجيز التابع التالي:

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

- n: عدد المدن,
- u and v: مصفوفتان بطول m. لكل $0 \leq i \leq m-1$, فإن $u[i]$ و $v[i]$ هما مدينتين متصلتين باستخدام الوصلة i.
- يتوجب على هذا التابع أن يعيد مصفوفة بطول $n-1$ تحتوي أرقام الوصلات الذهبية (بأي ترتيب عشوائي لها) .

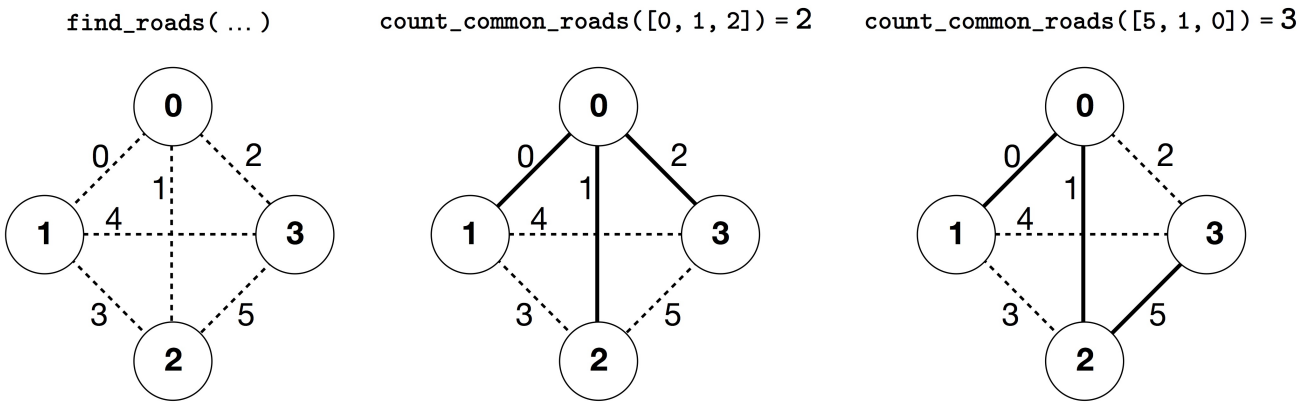
يستطيع حلك القيام بـ q استدعاء على الأكثر لتابع نظام التقييم التالي:

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

- r: مصفوفة بطول $n-1$ تشكل مجموعة ذهبية من الوصلات, وتحتوي على أرقام هذه الوصلات (بأي ترتيب).
- يعيد هذا التابع عدد الوصلات الملكية في r.

مثال

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



لدينا في هذا المثال 4 مدن و 6 وصلات. نقصد بالثنائية (a, b) الوصلة التي تصل بين المدينتين a و b. الوصلات مرقمة من 0 إلى 5 بالترتيب التالي من اليمين إلى اليسار: $(0,1)$, $(0,2)$, $(0,3)$, $(1,2)$, $(1,3)$, $(2,3)$. كل مجموعة ذهبية تحتوي على $n-1=3$ وصلة.

لنفترض بأن الوصلات الملكية هي الوصلات المرقمة 0, 1, 5, أي الوصلات $(0,1)$, $(0,2)$, $(2,3)$, عندها:

- `count_common_roads([0, 1, 2])`

هذا الاستدعاء هو للطرق المرقمة $0, 1$, 2, أي الطرق $(0,1)$, $(0,2)$, $(0,3)$, يعيد 2 لأن هناك طريقين ملكين بينهما.

- `count_common_roads([5, 1, 0])`

  يعيد 3. لأنه استدعاء يحتوي على كل الطرق الملكية.

يجب على التابع `find_roads` أن يعيد المصفوفة `[5, 1, 0]` أو أي مصفوفة أخرى بطول ثلاثة تحتوي على هذه الأرقام الثلاثة.

انتبه بأن الاستدعاءات التالية غير مسموحة:

- `count_common_roads([0, 1])`:

لأن طول المصفوفة r لا يساوي 3

- `count_common_roads([0, 1, 3])`:

هنا r ليست مجموعة ذهبية, وذلك لأنه من المستحيل الوصول من المدينة رقم 0 إلى المدينة رقم 3 باستخدام الوصلات $(0,1)$, $(0,2)$, $(1,2)$ فقط.

القيود

- $2 \leq n \leq 500$
- $n-1 \leq m \leq n(n-1)/2$
- $0 \leq u[i], v[i] \leq n-1$ (لكل $0 \leq i \leq m-1$)
- لكل $0 \leq i \leq m-1$, الوصلة i تصل بين مدينتين مختلفتين (i.e., $u[i] \neq v[i]$).
- هناك على الأكثر وصلة واحدة بين أي زوجين من المدن.
- من الممكن السفر بين أي مدينيتن باستخدام الوصلات.
- مجموعة كل الوصلات الملكية هي مجموعة ذهبية.
- يقوم التابع `find_roads` باستدعاء التابع `count_common_roads` على الأكثر q مرة. في كل استدعاء, يجب أن تكون المجموعة r عبارة عن مجموعة ذهبية.

المهمات الفرعية

1. (points 13) $n \leq 7$, $q = 30\,000$
2. (points 17) $n \leq 50$, $q = 30\,000$
3. (points 21) $n \leq 240$, $q = 30\,000$
4. (points 19) $q = 12000$ وهنالك وصلة بين أي زوجين من المدن
5. (points 30) $q = 8000$

Sample grader

The sample grader reads the input in the following format:

- line 1: $n \;\; m$
- line $2 + i$ (لكل $0 \leq i \leq m-1$): $u[i] \;\; v[i]$
- line $2 + m$: $s[0] \;\; s[1] \;\; \ldots \;\; s[n-2]$

هنا, $s[0], s[1], \ldots, s[n-2]$ هي أرقام الوصلات الملكية.

يطبع Sample grader `YES`, إذا قام التابع `find_roads` باستدعاء التابع `count_common_roads` على الأكثر $30\,000$ مرة, ويعيد المجموعة الصحيحة للوصلات الملكية. وإلا فإنه يطبع `NO`.

انتبه بأن التابع `count_common_roads` في sample grader لا يقوم بالتأكد فيما إذا كانت r تحقق جميع صفات المجموعة الذهبية. عوضاً عن ذلك, يقوم التابع بإعادة عدد الوصلات الملكية في المصفوفة r. على كل حال, إذا قام البرنامج باستدعاء التابع `count_common_roads` على مجموعة من الوصلات لا تشكل مجموعة ذهبية, فإن grading verdict سيكون هو 'Wrong Answer'

.

.

Technical note

The procedure `count_common_roads` in C++ and Pascal uses the *pass by reference* method for efficiency reasons. You can still call the procedure in the usual way. The grader is guaranteed not to change the value of r.