西默夫(Simurgh)

根據沙納瑪(Shahnameh)的古老波斯傳說，Zal，傳奇的波斯英雄，瘋狂地愛上喀布爾(Kabul)的公主Rudaba。當Zal向Rudaba求婚時，她的父親給Zal一個挑戰。

波斯有n個城市，編號從0到$n-1$，和m條雙向(two-way)道路，編號從0到$m-1$。每一條道路連接兩個不同的城市。任兩座城市最多只有一條的道路所連接。某些道路為被皇室成員所使用的*皇家道路* (*royal roads*)，而且這些皇家道路是被保密而不為外界所知。Zal的任務是要找出哪些道路是皇家道路。

Zal有一張標有波斯所有城市和道路的地圖。他不知道哪些道路是皇家道路，但他可求助於西默夫(Simurgh)，一隻仁慈的神鳥，也是Zal的守護神。無論如何，西默夫不想直接透露哪幾條道路是皇家道路。退而求其次，西默夫告知Zal所有皇家道路的集合是一個*黃金集合* (*golden set*)。黃金集合是一個道路的集合且滿足下列條件:

- 此集合*剛好*(*exactly*)有$n-1$條道路，且
- 對於任兩座城市，從一座城市可經由此集合的一些道路抵達另一座城市。

此外，Zal可以問西默夫一些問題。對於每個問題:

1. Zal 可選擇一個*黃金* (*golden*) 道路集合，然後
2. 西默夫告訴Zal ，在他所選的黃金集合裡有多少條道路是皇家道路。

你的程式可藉由問西默夫不超過q個問題，來幫忙Zal找到皇家道路的集合。評分程式(grader)將扮演西默夫的角色。

實作細節

你應實作下列程序(procedure):

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

- n: 城市的個數，
- u 和 v: 長度為m的陣列。對所有 $0 \leq i \leq m-1$, $u[i]$ 和 $v[i]$ 是被道路i所連接的城市。
- 此程序應該回傳一個長度為$n-1$的陣列包含皇家道路的編號(任意順序皆可)。

你的解答應呼叫下列評分程序(grader procedure)不超過q次:

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

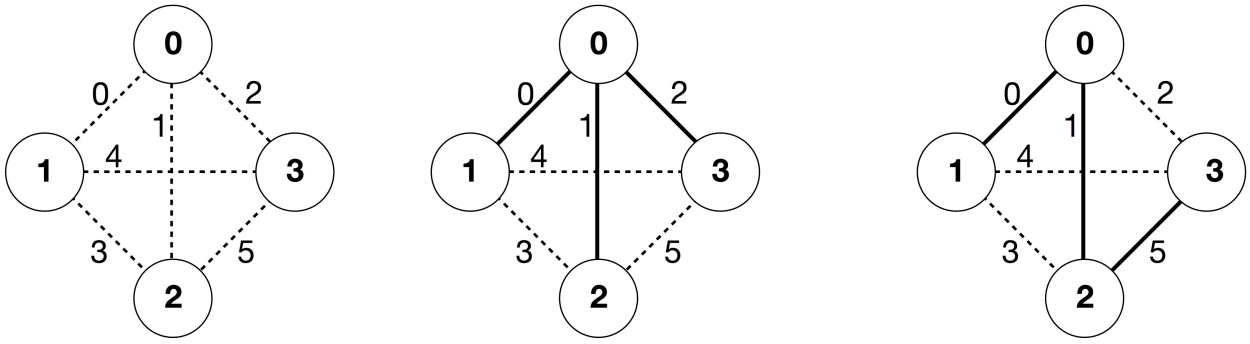
- r: 長度為$n-1$的陣列包含黃金集合裡面道路的編號(任意順序皆可)。.

- 此程序回傳r陣列中皇家道路的個數。

範例

```
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`



此範例有4座城市和6條道路。 (a, b)表示一條連接城市a和城市b的道路。 道路編號從0到5以下列順序出現: $(0, 1)$, $(0, 2)$, $(0, 3)$, $(1, 2)$, $(1, 3)$, 和 $(2, 3)$。每個黃金集合有$n - 1 = 3$條道路。

假設皇家道路編號為0, 1, 和 5, 亦即道路$(0, 1)$, $(0, 2)$, 和 $(2, 3)$, 且程式執行下列呼叫:

- `count_common_roads([0, 1, 2])` 回傳 2。此詢問(query)是關於道路編號$0, 1$, 和 2, 亦即道路 $(0, 1)$, $(0, 2)$ 和 $(0, 3)$。三條道路中有兩條是皇家道路。
- `count_common_roads([5, 1, 0])` 回傳 3。 此詢問是包含了所有皇家道路的集合。

程序 `find_roads` 應回傳 `[5, 1, 0]` 或任何長度為3且包含此三個元素的陣列。

注意下列呼叫是不被允許的:

- `count_common_roads([0, 1])`: 此處r的長度不等於3。
- `count_common_roads([0, 1, 3])`: 此處 r 不是描述黃金集合，因為要從城市0 到達城市 3 只藉由 $(0, 1)$, $(0, 2)$, $(1, 2)$是不可能的.

限制條件(Constraints)

- $2 \le n \le 500$
- $n - 1 \le m \le n(n - 1)/2$
- $0 \le u[i], v[i] \le n - 1$ (對於所有 $0 \le i \le m - 1$)
- 對於所有$0 \le i \le m - 1$, 道路編號i所連接的兩個不同的城市 (亦即$u[i] \ne v[i]$)。
- 任兩座城市最多被一條道路所連接。
- 一定可以經由道路來遊歷(travel)任兩個城市。
- 所有皇家道路的集合是黃金集合。
- `find_roads` 應呼叫 `count_common_roads` 不超過 q 次。 在每次呼叫中，被陣列r所指明的道路集合需是黃金集合。

Subtasks

1. (13 points) $n \leq 7, q = 30\,000$
2. (17 points) $n \leq 50, q = 30\,000$
3. (21 points) $n \leq 240, q = 30\,000$
4. (19 points) $q = 12000$ 且任兩座城市都有道路連接。
5. (30 points) $q = 8000$

範例評分程式(Sample grader)

範例評分程式讀讀取下列格式的輸入:

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

此處 $s[0], s[1], \ldots, s[n - 2]$ 是皇家道路的編號。

範例評分程式輸出 `YES`, 假如 `find_roads` 呼叫 `count_common_roads` 不超過 $30\,000$次，且回傳正確的皇家道路的集合。否則，輸出 `NO`。

請提防程序 `count_common_roads` 在範例評分程式中不會去檢查是否 r有黃金集合的所有性質。取而代之的是此程序會計算並回傳在陣列r中皇家道路編號的個數。 無論如何，假如你上傳的程式使用不是描述黃金集合的編號集合去呼叫`count_common_roads`，則評分程式的判定將是'Wrong Answer'。

技術注意事項(Technical note)

為求效率，程序 `count_common_roads` 在 C++ and Pascal 使用 *call by reference* 的方法。但是，你仍然可以用平常的方法呼叫程序。此外，評分程式保證不會改變 r值。