Simurgh

Σύμφωνα με τον αρχαίο Περσικό μύθο της Shahnameh, ο θρυλικός ήρωας Zal ήταν τρελά ερωτευμένος με την Rubala, την πριγκίπισσα της Kabul. Όταν ο Zal ζήτησε την Rubala σε γάμο, ο πατέρας της του ανέθεσε μια αποστολή.

Στην Περσία υπάρχουν n πόλεις, αριθμημένες από 0 μέχρι $n-1$, και m δρόμοι διπλής κατεύθυνσης, αριθμημένοι από 0 μέχρι $m-1$. Κάθε δρόμος ενώνει ένα ζεύγος διαφορετικών πόλεων. Κάθε ζεύγος πόλεων συνδέεται το πολύ από έναν δρόμο. Ορισμένοι από τους δρόμους είναι *βασιλικοί δρόμοι* τους οποίους χρησιμοποιεί η βασιλική οικογένεια. Η αποστολή του Zal είναι να ανακαλύψει ποιοι από τους δρόμους είναι βασιλικοί.

Ο Zal έχει έναν χάρτη με όλες τις πόλεις και τους δρόμους της Περσίας. Δεν γνωρίζει ποιοι από τους δρόμους είναι βασιλικοί αλλά μπορεί να ζητήσει βοήθεια από την Simurgh, το καλοκάγαθο μυθικό πουλί, που είναι η προστάτιδα του Zal. Ωστόσο, η Simurgh δεν θέλει να αποκαλύψει άμεσα το σύνολο των βασιλικών δρόμων. Αντιθέτως, πληροφορεί τον Zal ότι το σύνολο όλων των βασιλικών δρόμων είναι ένα *χρυσό σύνολο*. Ένα συνολο δρόμων είναι χρυσό σύνολο αν και μόνο αν:

- έχει *ακριβώς* $n-1$ δρομους και
- για κάθε ζεύγος πόλεων, είναι δυνατόν να φτάσει κανείς από την μία πόλη στην άλλη, ταξιδεύοντας μόνο από τους δρόμους αυτού του συνόλου.

Επιπλέον, ο Zal μπορεί να κάνει στην Simurgh κάποιες ερωτήσεις. Για κάθε ερώτηση:

1. Ο Zal επιλέγει ένα *χρυσό* σύνολο δρόμων και μετά
2. Η Simurgh απαντά στον Zal πόσοι από τους δρόμους αυτού του χρυσού συνόλου είναι βασιλικοί δρόμοι.

Το πρόγραμμά σας πρέπει να βοηθήσει τον Zal να βρει το σύνολο των βασιλικών δρόμων, ρωτώντας την Simurgh το πολύ q ερωτήσεις. Ο βαθμολογητής θα παίξει τον ρόλο της Simurgh.

Λεπτομέρειες υλοποίησης

Πρέπει να υλοποιήσετε την παρακάτω διαδικασία:

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

- n: το πλήθος των πόλεων.
- u και v: πίνακες μεγέθους m. Για κάθε $0 \leq i \leq m-1$, οι $u[i]$ και $v[i]$ είναι οι πόλεις που συνδέονται με τον δρόμο i.

- Η διαδικασία πρέπει να επιστρέφει έναν πίνακα μεγέθους $n-1$, ο οποίος να περιέχει τους αριθμούς των βασιλικών πόλεων (σε οποιαδήποτε σειρά).

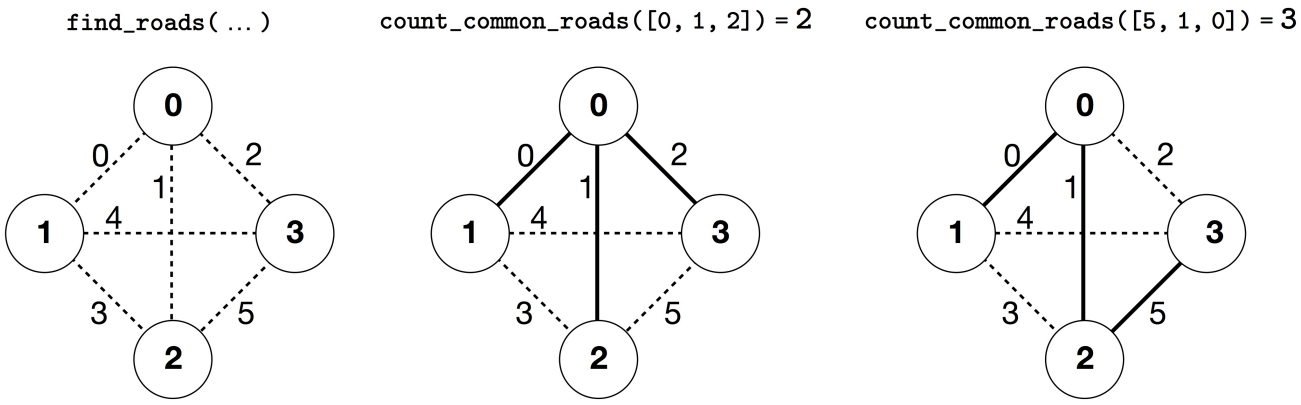
Η λύση σας πρέπει να κάνει το πολύ q κλήσεις της ακόλουθης διαδικασίας του βαθμολογητή:

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

- r: πίνακας μεγέθους $n-1$, ο οποίος περιέχει τους αριθμούς των πόλεων ενός χρυσού συνόλου (σε οποιαδήποτε σειρά).
- Αυτή η διαδικασία επιστρέφει το πλήθος των βασιλικών δρόμων που υπάρχουν στον πίνακα r.

Παράδειγμα

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



Σε αυτό το παράδειγμα υπάρχουν 4 πόλεις και 6 δρόμοι. Σημειώνουμε με (a, b) έναν δρόμο που συνδέει τις πόλεις a και b. Οι δρόμοι είναι αριθμημένοι από το 0 μέχρι το 5 με την ακόλουθη σειρά: $(0, 1)$, $(0, 2)$, $(0, 3)$, $(1, 2)$, $(1, 3)$, και $(2, 3)$. Κάθε χρυσό σύνολο έχει $n-1 = 3$ δρόμους.

Έστω ότι οι βασιλικοί δρόμοι είναι οι δρόμοι με αριθμούς 0, 1 και 5, δηλαδή οι δρόμοι $(0, 1)$, $(0, 2)$ και $(2, 3)$. Τότε:

- `count_common_roads([0, 1, 2])` επιστρέφει 2. Αυτό το ερώτημα αφορά τους δρόμους 0, 1 και 2, δηλαδή τους δρόμους $(0, 1)$, $(0, 2)$ και $(0, 3)$. Δύο από αυτούς είναι βασιλικοί δρόμοι.
- `count_common_roads([5, 1, 0])` επιστρέφει 3. Αυτό το ερώτημα αφορά ολόκληρο το σύνολο των βασιλικών δρόμων.

Η διαδικασία `find_roads` πρέπει να επιστρέψει `[5, 1, 0]` ή οποιονδήποτε πίνακα μεγέθους 3 που να περιέχει αυτά τα τρία στοιχεία.

Να σημειωθεί ότι οι ακόλουθες κλησεις δεν επιτρέπονται:

- `count_common_roads([0, 1])`: το μέγεθος του r δεν είναι 3.

- `count_common_roads([0, 1, 3])`: εδώ ο πίνακας r δεν περιγράφει ένα χρυσό σύνολο, γιατί είναι αδύνατον να ταξιδέψει κανείς απο την πόλη 0 στην πόλη 3 χρησιμοποιώντας αποκλειστικά τους δρόμους $(0, 1)$, $(0, 2)$, $(1, 2)$.

Περιορισμοί

- $2 \leq n \leq 500$
- $n - 1 \leq m \leq n(n-1)/2$
- $0 \leq u[i], v[i] \leq n - 1$ (για κάθε $0 \leq i \leq m - 1$)
- Για κάθε $0 \leq i \leq m - 1$, ο δρόμος i συνδέει δύο διαφορετικές πόλεις (δηλαδή, $u[i] \neq v[i]$).
- Υπάρχει το πολύ ένας δρόμος μεταξύ κάθε κάθε ζεύγους πόλεων.
- Χρησιμοποιώντας τους δρόμους μπορεί κανείς να ταξιδεψει μεταξύ οποιουδήποτε ζεύγους πόλεων.
- Το σύνολο όλων των βασιλικών δρόμων είναι ένα χρυσό σύνολο.
- Η `find_roads` πρέπει να καλέι την `count_common_roads` το πολύ q φορές. Σε κάθε κλήση, το σύνολο των δρόμων που σημειώνεται από τον πίνακα r πρέπει να είναι ένα χρυσό σύνολο.

Υποπροβλήματα

1. (13 βαθμοί) $n \leq 7$, $q = 30\,000$
2. (17 βαθμοί) $n \leq 50$, $q = 30\,000$
3. (21 βαθμοί) $n \leq 240$, $q = 30\,000$
4. (19 βαθμοί) $q = 12\,000$ και υπάρχει δρόμος που ενώνει κάθε ζεύγος πόλεων
5. (30 βαθμοί) $q = 8000$

Υπόδειγμα βαθμολογητή

Ο βαθμολογητής που σας δίνεται ως υπόδειγμα, διαβάζει την είσοδο με την παρακάτω μορφή:

- γραμμή 1: n m
- γραμμή $2 + i$ (για κάθε $0 \leq i \leq m - 1$): $u[i]$ $v[i]$
- γραμμή $2 + m$: $s[0]$ $s[1]$ $\ldots$ $s[n-2]$

Εδώ οι, $s[0], s[1], \ldots, s[n-2]$ είναι οι αριθμοί των βασιλικών δρόμων.

Ο βαθμολογητής τυπώνει `YES`, αν η `find_roads` καλέι την `count_common_roads` το πολύ $30\,000$ φορές και επιστρέφει το σωστό σύνολο με τους βασιλικούς δρόμους. Αλλιώς, τυπώνει `NO`.

Να σημειωθεί ότι η διαδικασία `count_common_roads` στο υπόδειγμα βαθμολογητή δεν ελέγχει αν ο πίνακας r έχει όλες τις ιδιότητες ενός χρυσού συνόλου. Αντιθέτως, μετράει και επιστρέφει το πλήθος των βασιλικών δρόμων που υπάρχουν στον πίνακα r. Ωστόσο, αν το πρόγραμμά σας καλέσει την `count_common_roads` με ένα σύνολο αριθμών που δεν περιγράφουν ένα χρυσό σύνολο θα πάρετε απάντηση 'Wrong Answer'.

Τεχνική σημείωση

Η διαδικασία `count_common_roads` στην C++ και στην Pascal χρησιμοποιεί *πέρασμα κατ' αναφορά* (pass by reference) για λόγους αποδοτικότητας. Μπορείτε να καλέσετε τη διαδικασία με τον συνηθισμένο τρόπο. Ο βαθμολογητής δεν πρόκειται να αλλάξει την τιμή του r.