სიმურგი

"შაჰნამე"-ში მოცემული ძველი სპარსული ლეგენდის მიხედვით, ლეგენდარულ სპარსელ გმირს, ზაალს უგონოდ უყვარდა ქაბულის პრინცესა რუდაბა. როდესაც ზაალმა ხელი სთხოვა რუდაბას, მამამისმა მას ერთი ამოცანის გადაწყვეტა დაავალა.

სპარსეთში n ცალი ქალაქია, რომლებიც გადანომრილია 0-დან to $(n-1)$-მდე. ქალაქებს აკავშირებს m ორმხრივი გზა, რომლებიც გადანომრილია 0-დან $(m-1)$-მდე. ყოველი გზა აერთებს ორ განსხვავებულ ქალაქს. ქალაქების ყოველი წყვილი შეერთებულია არაუმეტეს ერთი გზით. ზოგიერთი გზა წარმოადგენს *სამეფო გზას*, რადგან ისინი გამოიყენება სამეფო ოჯახის გადასაადგილებლად და მათი არსებობა გასაიდუმლოებულია. ზაალმა უნდა დაადგინოს, რომელია სამეფო გზები.

ზაალს აქვს რუკა, რომელზეც გამოსახულია სპარსეთის ყველა ქალაქი და ყველა გზა. მან არ იცის, ამ გზებიდან რომელია სამეფო გზები, მაგრამ შეუძლია დახმარება სთხოვოს კეთილ მითიურ ფრინველს - სიმურგს, რომელიც ზაალის მფარველია. მაგრამ სიმურგს არ სურს, პირდაპირ დაასახელოს სამეფო გზების სიმრავლე. ამის ნაცვლად, ის ეუბნება ზაალს, რომ სამეფო გზების ერთობლიობა წარმოადგენს *ოქროს სიმრავლეს*. გზათა ერთობლიობა წარმოადგენს ოქროს სიმრავლეს მაშინ და მხოლოდ მაშინ:

- ის შეიცავს *ზუსტად* $n-1$ გზას, და
- ქალაქთა ნებისმიერი წყვილისათვის შესაძლებელია ერთი მათგანიდან მეორეში მისვლა, მხოლოდ ამ სიმრავლეში შემავალი ქალაქების გავლით.

პასუხის გამოსაცნობად, ზაალს შეუძლია დაუსვას შეკითხვები სიმურგს. თითოეულ შეკითხვაში:

1. ზაალმა უნდა აარჩიოს გზათა *ოქროს* სიმრავლე, და
2. სიმურგმა უნდა უთხრას ზაალს, რამდენ სამეფო გზას შეიცავს დასახელებული ოქროს სიმრავლე.

თქვენი პროგრამა უნდა დაეხმაროს ზაალს სამეფო გზების სიმრავლის დადგენაში სიმურგისათვის არაუმეტეს q შეკითხვის დასმით. სიმურგის როლს გრადერი შეასრულებს.

იმპლემენტაციის დეტალები

თქვენ უნდა მოახდინოთ შემდეგი ფუნქციის იმპლემენტაცია:

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

- n: ქალაქების რაოდენობა,
- u და v: m სიგრძის მქონე მასივები. ყოველი $0 \leq i \leq m-1$, $u[i]$ და $v[i]$ წარმოადგენენ ქალაქებს, რომლებსაც აერთებს i-ური გზა.
- ფუნქციამ უნდა დააბრუნოს $n-1$ სიგრძის მასივი, რომელიც უნდა შეიცავდეს სამეფო გზების ნომრებს (ნებისმიერი თანმიმდევრობით).

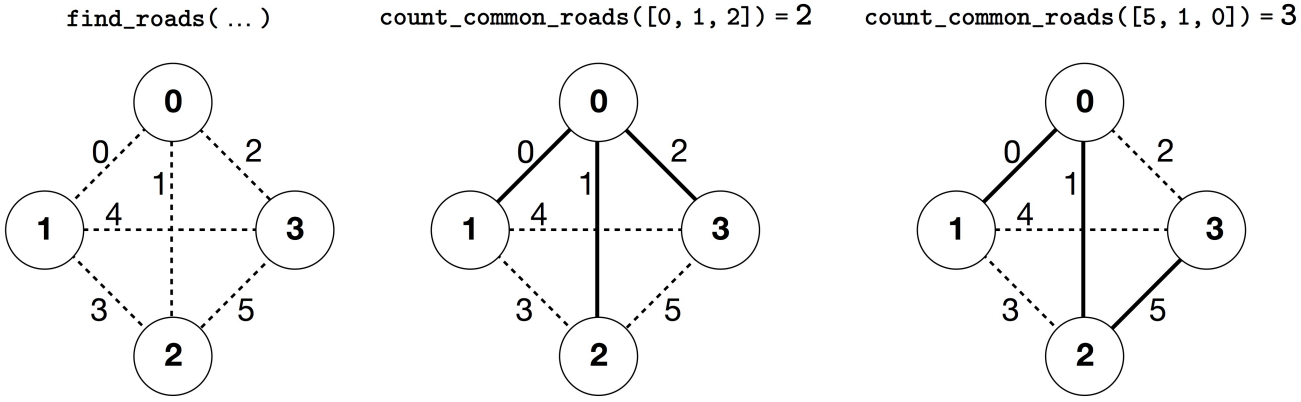
თქვენმა ამოხსნამ უნდა მოახდინოს არაუმეტეს q გამოძახება გრადერის შემდეგი ფუნქციისა:

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

- r: $n-1$ სიგრძის მასივი, რომელიც წარმოადგენს გზათა ოქროს სიმრავლეს (ნებისმიერი თანმიმდევრობით).
- ფუნქცია დააბრუნებს სამეფო გზების რაოდენობას r-დან.

მაგალითი

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



მოცემულ მაგალითში 4 ქალაქი და 6 გზაა. (a, b)-თი აღვნიშნავთ a და b ქალაქების დამაკავშირებელ გზას. გზები გადანომრილია 0-დან 5-მდე შემდეგი თანმიმდევრობით: $(0, 1)$, $(0, 2)$, $(0, 3)$, $(1, 2)$, $(1, 3)$, და $(2, 3)$. ყოველი ოქროს სიმრავლე შეიცავს $n-1=3$ გზას..

ვთქვათ, სამეფო გზების ნომრებია 0, 1, და 5, მაშინ ეს გზებია $(0, 1)$, $(0, 2)$, და $(2, 3)$, და პროგრამა აკეთებს შემდეგ გამოძახებებს:

- `count_common_roads([0, 1, 2])` დააბრუნებს 2. ეს შეკითხვა ეხება გზებს ნომრებით $0, 1,$ და 2. ეს გზებია $(0, 1)$, $(0, 2)$ და $(0, 3)$. ამ გზებიდან ორი სამეფო გზაა.
- `count_common_roads([5, 1, 0])` დააბრუნებს 3. ეს შეკითხვა ეხება ყველა

სამეფო გზისგან შედგენილ სიმრავლეს.

ფუნქციამ `find_roads` უნდა დააბრუნოს `[5, 1, 0]` ან 3 სიგრძის სხვა მასივი იგივე სამი ელემენტით.

შევნიშნოთ, რომ დაუშვებელია შემდეგი სახის გამოძახებები:

- `count_common_roads([0, 1])`: აქ r-ის სიგრძე არ არის 3.
- `count_common_roads([0, 1, 3])`: აქ r არ წარმოადგენს ოქროს სიმრავლეს, რადგან შეუძლებელია 0 ქალაქიდან 3 ქალაქამდე მიღწევა მხოლოდ $(0, 1)$, $(0, 2)$, $(1, 2)$ გზებით.

შეზღუდვები

- $2 \leq n \leq 500$
- $n - 1 \leq m \leq n(n-1)/2$
- $0 \leq u[i], v[i] \leq n - 1$ (ყოველი $0 \leq i \leq m - 1$)
- ყოველი $0 \leq i \leq m - 1$, სადაც i-ური გზა აკავშირებს ორ განსხვავებულ ქალაქს (ანუ, $u[i] \neq v[i]$).
- ქალაქთა ნებისმიერ წყვილს აკავშირებს არაუმეტეს ერთი გზისა.
- შესაძლებელია მოძრაობა ქალაქთა ნებისმიერ წყვილს შორის მოცემული გზებით
- ყველა სამეფო გზის სიმრავლე წარმოადგენს ოქროს სიმრავლეს.
- `find_roads`-ს შეუძლია `count_common_roads`-ის გამოძახება არაუმეტეს q-ჯერ. ყოველი გამოძახებისას r მასივში შემავალი ქალაქთა ნომრები უნდა წარმოადგენდნენ ოქროს სიმრავლეს.

ქვეამოცანები

1. (13 ქულა) $n \leq 7$, $q = 30\,000$
2. (17ქულა) $n \leq 50$, $q = 30\,000$
3. (21 ქულა) $n \leq 240$, $q = 30\,000$
4. (19 ქულა) $q = 12000$ და არსებობს გზა ქალაქთა ნებისმიერ წყვილს შორის.
5. (30 ქულა) $8000 \leq q \leq 30\,000$

სანიმუშო გრადერი

სანიმუშო გრადერი კითხულობს შესატან მონაცემებს შემდეგი ფორმატით:

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

აქ $s[0], s[1], \ldots, s[n-2]$ არის სამეფო გზების ნომრები.

სანიმუშო გრადერი გამოიტანს `YES`-ს, თუ `find_roads` გამოიძახებს

`count_common_roads`-ს არაუმეტეს 30 000-ჯერ და დააბრუნებს სამეფო გზების კორექტულ სიმრავლეს. წინააღმდეგ შემთხვევაში გამოიტანს `NO`.

გაითვალისწინეთ, რომ `count_common_roads` სანიმუშო გრადერში არ ამოწმებს, წარმოადგენს თუ არა r ოქროს სიმრავლეს. ის დაითვლის და დააბრუნებს სამეფო გზების რაოდენობას r მასივში. თუმცა პროგრამის ტესტირებისას ანალოგიურ სიტუაციაში გრადერის ვერდიქტი იქნება 'Wrong Answer'.

ტექნიკური შენიშვნა

ფუნქცია `count_common_roads` C++ და Pascal ენებზე იყენებს *call by reference* მეთოდს ეფექტურობის მიზნით. თქვენ შეგიძლიათ გამოიძახოთ ფუნქცია ჩვეულებისამებრ. გრადერისაგან გარანტირებულია, რომ არ შეცვლის r-ის მნიშვნელობას.