თამაში

ჯიან-ჯი პატარა ბიჭია, რომელსაც უყვარს თავსატეხები. როცა მას ეკითხებიან რაიმეს, ის ამჯობინებს თამაშით პასუხებს, ვიდრე უპასუხოს შეკითხვას პირდაპირ.
ჯიან-ჯის შეხვდა მეგობარი მეი-იუ, რომელსაც მოუყვა თუ როგორაა აწყობილი ავიაფრენები ტაივანში.
ტაივანში არის n ქალაქი, რომლებიც გადანომრილია 0-დან (n-1)–მდე. ზოგიერთ მათგანს შორის არსებობს ავიარეისები. ყოველი ავიარეისი ორმხრივია.

მეი-იუ შეეკითხა ჯიან-ჯის, მართალია თუ არა, რომ შესაძლებელია ნებისმიერი ქალაქიდან მივიდეს ნებისმიერ მხოლოდ ავიარეისებით (პირდაპირი ან არაპირდაპირი გზებით). ჯიან-ჯის არ სურს პირდაპირ უპასუხოს და ამის მაგივრად შესთავაზა შემდეგი თამაში: მეი-იუს შეუძლია მისცეს მას შეკითხვა შემდეგი ფორმით "შეიძლება თუ არა X ქალაქიდან y ქალაქამდე პირდაპირი გზით მისვლა?" და ჯიან-ჯი მაშინვე უპასუხებს დასმულ შეკითხვას. მეი-იუმ უნდა დასვას შეკითხვა ყოველ წყვილზე მხოლოდ ერთხელ, სულ - $r = n(n-1)/2$ შეკითხვა. მეი-იუ მოიგებს თამაშს თუ პირველი i შეკითხვის შემდეგ, რაიმე $i < r$, მას შეუძლია უპასუხოს შეკითხვას, მართალია თუ არა, რომ ავიარეისების ქსელი ბმულია. რაც ნიშნავს, რომ ნებისმიერი ქალაქიდან ნებისმიერში შეიძლება მისვლა მხოლოდ ავიარეისების გამოყენებით (პირდაპირი ან არაპირდაპირით). თუ მას დასჭირდება ყველა r შეკითხვა, მაშინ მოიგებს ჯიან-ჯი.

იმისათვის, რომ თამაში უფრო საინტერესო გახდეს, მოილაპარაკეს, რომ დაივიწყონ ტაივანში არსებული ავიასისტემა და მოიფიქრონ თამაშის პროცესში თავისი, მეი-იუს წინა შეკითხვების მიხედვით. თქვენ უნდა დაწეროთ პროგრამა, რომელიც დაეხმარება ჯიან-ჯის მოიგოს თამაში თავის მიერ მიცემული პასუხებით.

მაგალითები

ჩვენ განვიხილავთ თამაშის წესებს შემდეგი სამი მაგალითის მიხედვით. ყოველ მათგანში $n = 4$ ქალაქია და $r = 6$ კითხვა-პასუხის რაუნდი.

პირველ მაგალითში (ცხრილი ქვემოთაა) ჯიან-ჯი აგებს იმიტომ, რომ მე-4 რაუნდის შემდეგ მეი-იუ უკვე დარწმუნებულია რომ ნებისმიერ ორ ქალაქს შორის შეიძლება იმგზავრო მხოლოდ ავიარეისებით, მიუხედავად იმისა როგორი იქნება პასუხი 5 და 6 კითხვაზე.

რაუნდი	კითხვა	პასუხი
1	0, 1	yes
2	3, 0	yes
3	1, 2	no

რაუნდი	კითხვა	პასუხი
4	0, 2	yes
-----	--------	------
5	3, 1	no
6	2, 3	no

შემდეგ მაგალითში მეი-იუს მე-3 რაუნდის შემდეგ შეუძლია დაამტკიცოს, რომ როგორი პასუხიც არ უნდა მისცეს ჯიან-ჯიმ დარჩენილ კითხვებზე არ შეიძლება მივაღწიოთ 1 ქალაქიდან 2 ქალაქამდე და ჯიან-ჯი ისევ წააგებს.

რაუნდი	კითხვა	პასუხი
1	0, 3	no
2	2, 0	no
3	0, 1	no
-----	--------	------
4	1, 2	yes
5	1, 3	yes
6	2, 3	yes

ბოლო მაგალითში მეი-იუს არ შეუძლია პასუხის განსაზღვრა, სანამ არ მიიღებს პასუხებს 6-ვე კითხვაზე, ამიტომ ჯიან-ჯი იგებს. კერძოდ, რადგანაც ჯიან-ჯიმ ბოლო კითხვაზე უპასუხა *yes* მისვლა ნებისმიერი ქალაქიდან ნებისმიერ ქალაქამდე შესაძლებელია. თუ იგი უპასუხებს *no* მაშინ ეს შეუძლებელია.

რაუნდი	კითხვა	პასუხი
1	0, 3	no
2	1, 0	yes
3	0, 2	no
4	3, 1	yes
5	1, 2	no
6	2, 3	yes

ამოცანა

დაწერეთ პროგრამა, რომელიც დაეხმარება ჯიან-ჯის მოიგოს თამაში. მიაქციეთ ყურადღება, რომ მეი-იუმ და ჯიან-ჯიმ ერთმანეთის სტრატეგია არ იციან. მეი-იუს შეუძლია კითხვები დასვას ნებისმიერი მიმდევრობით და ჯიან-ჯიმ უნდა უპასუხოს მაშინვე - არ დაუცადოს დანარჩენ შეკითხვებს. თქვენ რეალიზება უნდა გაუკეთოთ ორ ფუნქციას:

- `initialize(n)` -- თავიდან გამოძახებული იქნება `initialize`. პარამეტრი n არის ქალაქების რაოდენობა.

- `hasEdge(u, v)` -- შემდეგ გამოძახებული იქნება `hasEdge` $r = n(n-1)/2$-ჯერ. ეს გამოძახებები წარმოადგენენ მეი-იუს შეკითხვებს იმ მიმდევრობით, რა მიმდევრობითაც ის მათ სვამს. თქვენ უნდა უპასუხოთ, არსებობს თუ არა პირდაპირი რეისი u და v ქალაქებს შორის. კერძოდ, დაბრუნებული მნიშვნელობა უნდა იყოს 1, თუ პირდაპირი რეისი არსებობს, ან 0 – წინააღმდეგ შემთხვევაში.

ქვეამოცანები

ქვეამოცანა	ქულა	n
1	15	$n = 4$
2	27	$4 \leq n \leq 80$
3	58	$4 \leq n \leq 1500$

რეალიზაციის დეტალები.

თქვენ უნდა გააგზავნოთ ზუსტად 1 ფაილი- `game.c`, `game.cpp` ან `game.pas`. ამ ფაილში რეალიზებული უნდა იქნეს პროცედურები ქვემოთ აღწერილი პროტოტიპებით.

C/C++ programs

```
void initialize(int n);
int hasEdge(int u, int v);
```

Pascal programs

```
procedure initialize(n: longint);
function hasEdge(u, v: longint): longint;
```

გრადერის მაგალითი.

გრადერის მაგალითს აქვს მონაცემთა შეტანის შემდეგი ფორმატი:

- ხაზი 1: `n`
- შემდეგი r ხაზი: ყოველი ხაზი შეიცავს ორ მთელს `u` და`v` ,რომლებიც აღწერენ კითხვას ორ u და v ქალაქს შორის.