Trò chơi

Jian-Jia là cậu bé yêu thích các trò chơi. Khi được yêu cầu trả lời một câu hỏi, cậu thích chơi một trò chơi hơn thay vì trả lời trực tiếp câu hỏi đó. Jian-Jia gặp Mei-Yu và kể cho cô bạn về mạng lưới chuyến bay ở Đài Loan. Có n thành phố ở Đài Loan (được đánh số từ 0, ..., $n-1$), một số thành phố được nối với nhau bởi các chuyến bay. Mỗi chuyến bay nối hai thành phố và có thể đi theo cả hai chiều.

Mei-Yu hỏi Jian-Jia liệu có thể di chuyển giữa hai thành phố bất kỳ bằng máy bay (hoặc là trực tiếp hoặc gián tiếp) hay không. Jian-Jia không muốn tiết lộ câu trả lời mà thay vì điều đó đề nghị chơi một trò chơi. Mei-Yu có thể hỏi cậu các câu hỏi có dạng: "Giữa hai thành phố x và y có chuyến bay *trực tiếp* hay không?" và Jian-Jia sẽ trả lời các câu hỏi ngay lập tức. Mei-Yu sẽ đặt câu hỏi cho mỗi cặp thành phố đúng một lần, đưa ra tổng cộng là $r = n(n-1)/2$ câu hỏi. Mei-Yu sẽ giành phần thắng nếu sau khi nhận được câu trả lời cho i câu hỏi đầu tiên $i < r$, cô có thể khẳng định được mạng có là liên thông hay không, nghĩa là có thể di chuyển giữa hai thành phố bất kỳ bằng các chuyến bay (hoặc là trực tiếp hoặc gián tiếp). Ngược lại, nếu cô cần hỏi tất cả r câu hỏi thì người giành phần thắng là Jian-Jia.

Để trò chơi được thú vị hơn cho Jian-Jia, hai bạn thỏa thuận với nhau rằng Jian-Jia có thể không sử dụng mạng lưới chuyến bay thực tế của Đài Loan và tạo ra mạng theo diễn biến của trò chơi, lựa chọn câu trả lời dựa trên các câu hỏi trước của Mei-Yu. Nhiệm vụ của bạn là giúp Jian-Jia giành phần thắng của trò chơi bằng cách đưa ra những trả lời cho các câu hỏi.

Ví dụ

Chúng ta giải thích quy tắc của trò chơi qua ba ví dụ. Mỗi ví dụ có $n = 4$ thành phố và $r = 6$ lượt hỏi và trả lời.

Trong ví dụ thứ nhất (như bảng dưới đây), Jian-Jia *thua* bởi vì sau lượt thứ 4, Mei-Yu biết chắc chắn rằng có thể di chuyển giữa hai thành phố bất kỳ bằng các chuyến bay, không cần đến câu trả lời cho câu hỏi thứ 5 hoặc thứ 6.

Lượt	Câu hỏi	Trả lời
1	0, 1	yes
2	3, 0	yes
3	1, 2	no
4	0, 2	yes
-----	--------	------
5	3, 1	no
6	2, 3	no

Trong ví dụ tiếp theo Mei-Yu có thể chỉ ra là *không thể* di chuyển giữa hai thành phố 0 và 1 bằng các

chuyến bay sau lượt thứ 3, không cần đến câu trả lời cho câu hỏi thứ 4, 5 hoặc 6 của Jian-Jia, do đó Jian-Jia lại thua.

Lượt	Câu hỏi	Trả lời
1	0, 3	no
2	2, 0	no
3	0, 1	no
-----	--------	------
4	1, 2	yes
5	1, 3	yes
6	2, 3	yes

Trong ví dụ cuối cùng, Mei-Yu không thể xác định liệu có cách di chuyển giữa hai thành phố bất kỳ bằng các chuyến bay trước khi tất cả sáu câu hỏi đều được trả lời, do đó Jian-Jia giành *phần thắng*. Cụ thể, bởi vì Jian-Jia đã trả lời *yes* cho câu hỏi cuối cùng (trong bảng dưới đây), khi đó có thể di chuyển giữa hai thành phố bất kỳ. Tuy nhiên, nếu Jian-Jia trả lời *no* cho câu hỏi cuối cùng thì điều đó là không thể.

Lượt	Câu hỏi	Trả lời
1	0, 3	no
2	1, 0	yes
3	0, 2	no
4	3, 1	yes
5	1, 2	no
6	2, 3	yes

Nhiệm vụ

Bạn hãy viết chương trình giúp Jian-Jia giành phần thắng trong trò chơi. Chú ý rằng đối với cả Mei-Yu lẫn Jian-Jia, người này không biết chiến lược chơi của người kia và ngược lại. Mei-Yu có thể hỏi các cặp thành phố theo thứ tự bất kỳ và Jian-Jia phải trả lời ngay lập tức, không được biết các câu sắp hỏi. Bạn cần cài đặt hai sau.

- `initialize(n)` -- Hàm `initialize` sẽ được gọi đầu tiên. Tham số n là số lượng thành phố.
- `hasEdge(u, v)` -- sau đó hàm `hasEdge` sẽ được gọi $r = n(n-1)/2$ lần. Các lần gọi này thể hiện cho các câu hỏi của Mei-Yu theo thứ tự mà cô ta thực hiện. Bạn phải trả lời có hay không có chuyến bay trực tiếp giữa hai thành phố u và v. Cụ thể, giá trị trả về phải là 1 nếu có chuyến bay trực tiếp, hay là 0 trong trường hợp ngược lại.

Subtasks

Mỗi subtask bao gồm nhiều lần chơi. Bạn chỉ được điểm cho subtask nếu chương trình của bạn thắng tất cả các lần chơi cho Jian-Jia.

subtask	points	n
1	15	$n = 4$
2	27	$4 \leq n \leq 80$
3	58	$4 \leq n \leq 1500$

Chi tiết cài đặt

Bạn phải nộp đúng một file được đặt tên `game.c`, `game.cpp` hoặc `game.pas`. File này cài đặt chương trình con như mô tả ở trên sử dụng các đặc tả dưới đây.

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;
```

Sample grader

Sample grader đọc dữ liệu vào theo khuôn dạng sau:

- Dòng 1: `n`
- r dòng tiếp theo: mỗi dòng chứa hai số nguyên `u` và `v` mô tả câu hỏi với hai thành phố u và v.