게임

지안지아는 게임하는 (퀴즈) 식으로 놀이를 만들기를 즐긴다. 즉, 지안지아는 질문을 받았을 때, 바로 정답을 알려 주는 것 보다는 퀴즈 식으로 놀이를 만들기를 좋아한다. 지안지아는 메이유라는 친구를 만나서 타이완의 여객항공 네트워크에 대해 말을 했다. 타이완에는 n 개의 도시가 있고 (0 부터 $n-1$ 까지 번호가 붙음), 이들 도시의 쌍들 중에는 여객기가 운항하는 쌍들이 있다. **운항한다**는 말은 두 도시 사이에 여객기가 양방향으로 **직접** (다른 도시를 거치지 않고) 다닌다는 말이다.

메이유가 지안지아에게 모든 쌍의 도시가 항공 네트워크를 통해 (직접 혹은 간접적으로) 연결이 되어 있는지 물어보았다. 지안지아는 이 질문에 대답을 바로 하는 대신, 게임을 하자고 제안했다. 메이유는 다음과 같은 형식의 질문만을 할 수 있다: "두 도시 x와 y 사이에 여객기가 (**직접**) 운항하는가?" 지안지아는 이런 식의 질문에는 바로 대답할 것이다. 메이유는 모든 쌍의 도시에 대해서 위와 같은 질문을 한번 씩 할 것이다. 그러면 질문의 수는 $r = n(n-1)/2$ 개가 될 것이다. 메이유가 r보다 작은 i개의 질문에 대한 대답을 들은 직후에, 전체 네트워크가 **연결되었는지 아닌지 결정**할 수 있게 되면 메이유가 이긴 것이다. 즉, 모든 도시의 쌍에 대해서 (직접 혹은 간접으로) 이동할 수 있는 경우인지, 아니면 어떤 도시의 쌍에 대해서는 이동이 불가능한 경우인지 알아내게 되면, 메이유가 이긴다. 반대로, 메이유가 r개의 질문을 전부 하고 대답을 들은 다음에야, 전체 네트워크의 연결 여부를 알아낼 수 있다면 지안지아가 이긴 것이다.

게임이 더 재미있도록 하기 위해서 실제 타이완의 항공 네트워크가 아닌 가상의 항공 네트워크를 가정하고, 메이유가 질문을 하는 것에 따라서 이전 질문에 대한 대답과는 모순이 되지 않는 한 네트워크의 구성을 만들어 가면서 대답을 할 수 있도록 하였다. 당신은 지안지아가 게임에서 이길 수 있도록 어떤 식으로 대답을 할지 도와주는 프로그램을 작성해야 한다.

예제

세가지 예제로 게임을 설명한다. 각 예제에는 $n = 4$ 개의 도시와 $r = 6$ 라운드의 질문과 대답이 있다.

다음 표에 있는 첫번째 예제에서는 지안지아가 지는데, 4 라운드의 질문 직후에 (이후 지안지아가 뭐라고 대답을 하든 간에) 메이유는 **네트워크가 연결되었다**는, 즉 모든 쌍의 도시 사이에 이동이 가능하다는 것을 알게 되기 때문이다.

라운드	질문	대답
1	0, 1	yes
2	3, 0	yes
3	1, 2	no
4	0, 2	yes
-----	--------	------
5	3, 1	no
6	2, 3	no

다음 예제에서 3라운드 직후에 메이유는 (이후 지안지아가 뭐라고 대답을 하든 간에) 도시 0과 도시 1은 연결되지 않았다는, 즉 **네트워크가 끊어졌다**는 것을 알게 되므로, 이 경우에도 지안지아는 진 것이다.

라운드	질문	대답
1	0, 3	no
2	2, 0	no
3	0, 1	no
-----	--------	------
4	1, 2	yes
5	1, 3	yes
6	2, 3	yes

아래 마지막 예에서 최종 라운드의 대답이 나오기 전에는 메이유가 네트워크가 연결되었는지 끊어졌는지 알 수가 없다. 그래서 이 경우는 지안지아가 이긴 것이다. 좀더 자세히 설명하면, 표에서 지안지아가 마지막 질문에 **yes** 라고 대답했기 때문에 모든 쌍의 도시가 연결되었다는 것이 **확정**되는 것이다. 만약 마지막 대답이 **no** 였다면 모든 쌍의 도시가 연결된 것은 아니게 된다.

라운드	질문	대답
1	0, 3	no
2	1, 0	yes
3	0, 2	no
4	3, 1	yes
5	1, 2	no
6	2, 3	yes

문제

지안지아가 게임에서 이길 수 있도록 도와주는 프로그램을 작성하시오. 메이유나 지안지아는 서로의 작전을 모른다고 가정해야 한다. 메이유가 질문을 하는 순서는 어떤 순서이든 가능하며, 지안지아는 (이후의 질문이 어떻게 될지 모르는 상태에서) 바로 대답을 해야 한다. 다음의 두 함수를 구현하시오.

- `initialize(n)` -- 이 함수가 최초로 호출된다. 파라미터 n 은 도시의 수이다.
- `hasEdge(u, v)` -- 이 함수가 $r(=n(n-1)/2)$번 호출된다. 이 호출들은 메이유의 질문들에 해당한다. 물론 호출되는 순서는 메이유가 질문하는 것과 동일하다. 두 도시 u 와 v 사이에 여객기가 직접 운항하는 지 대답하여야 한다. 여객기가 직접 운항하는 경우 1을 리턴하고, 아닌 경우 0을 리턴해야 한다.

부분문제

각 부분문제는 몇개의 게임들로 구성된다. 모든 게임을 다 이겨야 부분문제에 대한 점수를 받는다.

부분문제	점수	n
1	15	$n = 4$
2	27	$4 \leq n \leq 80$
3	58	$4 \leq n \leq 1,500$

구현 내용

단 하나의 파일을 제출한다. 파일의 이름은 `game.c`, `game.cpp`, `game.pas` 중 하나이다. 이 파일에는 위에서 설명한 함수(서브프로그램)들이 아래의 선언대로 구현되어 있어야 한다.

C/C++ 프로그램

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

Pascal 프로그램

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

Sample grader

Sample grader의 입력 양식은 다음과 같다.

- line 1: `n`
- the following r lines: 각 라인에는 두개의 정수 `u` 와 `v` 가 주어지는데, 두 도시 u 와 v 에 대한 질문이라는 뜻이다. 라인 순서와 질문 순서는 같다.