Game

Jian-Jia es un joven al que le gusta resolver acertijos. Cuando se le hace una pregunta , él prefiere ponerse a jugar en vez de responder directamente. Jian-Jia se encontró con su amiga Mei-Yu y se puso a hablar de la red de vuelos en Taiwan. Hay n ciudades en Taiwan (numeradas 0, ..., $n-1$), algunas de las cuales están conectadas por vuelos. Cada vuelo conecta dos ciudades y puede ser tomado en ambos sentidos.

Mei-Yu le pregunta a Jian-Jia si es posible ir entre cualquier par de ciudades por avión (directamente o indirectamente). Jian-Jia no quiere revelar la respuesta, en vez de esto sugirió un juego. Mei-Yu puede hacerle preguntas de la forma: "¿Están conectadas las ciudades x y y *directamente* con un viaje?" y Jian—Jia contestará tal pregunta de manera inmediata. Mei-Yu preguntará por cada par de ciudades, dando $r = n(n-1)/2$ preguntas en total. Mei-Yu gana el juego, si después de i preguntas para algún $i < r$, ella puede inferir si es posible o no viajar entre cualquier par de ciudades x y y por vuelos (o directamente o indirectamente). En otro caso, si ella necesita hacer todas las r preguntas, entonces Jin-Jia gana el juego.

Con la finalidad de hacer que el juego sea más divertido para Jian-Jia, los amigos acuerdan que él puede olvidarse de la red real de vuelos taiwanes e inventar la red conforme el juego progrese, escogiendo sus respuesta basandose en las preguntas previas de Mei-Yu. Su tarea es ayudar a Jian-Jia a ganar el juego, decidiendo cómo debería él responder las preguntas.

Examples

Explicaremos las reglas del juego con tres ejemplos. Cada ejemplo tiene $n = 4$ ciudades y $r = 6$ rondas de preguntas y respuestas.

En el primer ejemplo (la siguiente tabla), Jian-Jia *pierde* porque después de la ronda 4, Mei-Yu sabe con seguridad que uno puede viajar entre cualquier par de ciudades por vuelos, sin importar como Jian-Jia reponda las preguntas 5 o 6.

ronda	pregunta	respuesta
1	0, 1	si
2	3, 0	si
3	1, 2	no
4	0, 2	si
-----	--------	------
5	3, 1	no
6	2, 3	no

En el siguiente ejemplo Mei-Yu puede probar que después de la ronda 3 sin importar como responda Jian-Jia las preguntas 4, 5 o 6, uno *no puede viajar* entre las ciudades 0 y 1 por vuelos, por lo tanto

Jian-Jia pierde nuevamente.

ronda	pregunta	respuesta
1	0, 3	no
2	2, 0	no
3	0, 1	no
-----	--------	------
4	1, 2	si
5	1, 3	si
6	2, 3	si

En el ejemplo final Mei-Yu no puede determinar si él puede o no viajar entre dos ciudades cualesquiera por vuelos hasta que estén respondidas todas las seis preguntas, asi Jian-Jia gana el juego. Específicamente, como Jian-Jia respondió *si* a la última pregunta (en la siguiente tabla), entonces es posible viajar entre cualesquier par de ciudades. Sin embargo, si Jian-Jia hubiera respondido *no* en lugar de *si* a la última pregunta entonces sería imposible.

ronda	pregunta	respuesta
1	0, 3	no
2	1, 0	si
3	0, 2	no
4	3, 1	si
5	1, 2	no
6	2, 3	si

Task

Por favor, escriba un programa que ayude a Jian-Jia a ganar el juego. Note que ni Mei-Yu ni Jian-Jia conocen la estrategia del otro. Mei-Yu puede preguntar por pares de ciudades en cualquier orden y Jian-Jia debe reponder inmediatamente sin saber las preguntas siguientes. Usted necesita implementar las siguientes dos funciones.

- `initialize(n)` -- Llamaremos primero a su `initialize`. El paramétro n es el número de ciudades.
- `hasEdge(u, v)` -- Luego llamaremos `hasEdge` para $r = n(n-1)/2$ veces. Esos llamados representan las preguntas de Mei-Yu's, en el orden en que ella las hace. Usted debe responder si hay vuelo directo entra las ciudades u y v. Específicamente, el valor de retorno debe ser 1 si hay un vuelo directo y 0 en otro caso.

Subtasks

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

Implementation details

You have to submit exactly one file, called `game.c`, `game.cpp` or `game.pas`. This file implements the subprograms described above using the following signatures.

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

The sample grader reads the input in the following format:

- line 1: `n`
- the following r lines: each line contains two integers `u` and `v` that describe a question regarding cities u and v.