Igra

Andrej je dječak koji voli igre. Kada Andreju postavljaju pitanja, on radije igra igru zavlačenja umjesto da odgovora direktno. Jednog dana, Andrej je sreo svog druga Ivana i ispričao mu o tajvanskoj avionskoj mreži. U Tajvanu se nalazi n gradova (numerisanih brojevima 0, ..., $n-1$) i neki od njih su povezani avionskim linijama. Svaki let povezuje neka dva različita grada i njime se može letjeti u oba smjera.

Ivan je pitao Andreja da li je moguće putovati avionom između svaka dva grada (bilo direktno ili indirektno). Andrej nije želeo da mu otkrije odgovor već je predložio da igraju igru. Ivan može da mu postavlja isključivo pitanja oblika "Da li su gradovi x i y *direktno* povezani avionskom linijom?", a Andrej na ova pitanja odgovara odmah. Ivan će pitati pitanja za svaki par gradova tačno jednom, tj. postaviće ukupno $r = n(n-1)/2$ pitanja. Ivan pobjeđuje ako, posle dobijanja odgovora na prvih i pitanja za neko $i < r$, on može zaključiti da li je avionska mreža povezana, tj. da li je moguće putovati avionom između svaka dva grada (bilo direktno ili indirektno). U suprotnom, tj. ako mu je potrebno svih r pitanja, pobjeđuje Andrej.

Da bi igra bila zabavnija za Andreja, drugari su se složili da Andrej može zaboraviti pravu tajvansku avionsku mrežu i da mu je dopušteno da izmišlja ovu mrežu dok igra traje, birajući svoje odgovore na osnovu Ivanovih prethodnih pitanja.

Vaš zadatak je da pomognete Andreju da pobijedi tako što ćete odlučivati kako će on odgovarati na pitanja.

Primjeri

Objasnićemo detaljnije pravila igre uz pomoć tri primjera. Svaki primjer ima $n = 4$ gradova i $r = 6$ pitanja-odgovora.

U prvom primjeru (sljedeća tabela), Andrej *gubi* jer poslije četvrtog pitanja-odgovora, Ivan sa sigurnošću može zaključiti da je moguće putovati avionom između svaka dva grada, bez obzira na to kako Andrej odgovori na pitanja 5 ili 6.

redni broj	pitanje	odgovor
1	0, 1	da
2	3, 0	da
3	1, 2	ne
4	0, 2	da
-----	--------	------
5	3, 1	ne
6	2, 3	ne

U sljedećem primjeru, posle trećeg pitanja-odgovora Ivan može dokazati da bez obzira na to kako Andrej odgovori na pitanja 4, 5 i 6, *nije moguće* putovati avionom (ni direktno ni indirektno) između gradova 0 i 1, pa Andrej ponovo gubi.

redni broj	pitanje	odgovor
1	0, 3	ne
2	2, 0	ne
3	0, 1	ne
-----	--------	------
4	1, 2	da
5	1, 3	da
6	2, 3	da

U posljednjem primjeru Ivan ne može odrediti da li je moguće putovati avionom između svaka dva grada sve dok ne dobije odgovor na svih šest pitanja, pa Andrej *pobjeđuje*. Inače, kako je Andrej odgovorio *da* na posljednje pitanje (u sljedećoj tabeli), slijedi da je moguće putovati avionom između svaka dva grada. Međutim, da je Andrej odgovorio *ne* na posljednje pitanje, tada bi bilo nemoguće putovati avionom između svaka dva grada.

redni broj	pitanje	odgovor
1	0, 3	ne
2	1, 0	da
3	0, 2	ne
4	3, 1	da
5	1, 2	ne
6	2, 3	da

Zadatak

Napišite program koji pomaže Andreju da pobijedi. Obratite pažnju da ni Andrej ni Ivan ne znaju strategiju onog drugog. Ivan može postavljati pitanja za parove gradova u proizvoljnom redosljedu, dok Andrej mora na njih odgovarati odmah, ne znajući buduća pitanja. Potrebno je da implementirate sljedeće dvije funkcije.

- `initialize(n)` -- Prvo ćemo pozvati vaš `initialize`. Parametar n je broj gradova.
- `hasEdge(u, v)` -- Zatim ćemo pozvati `hasEdge` $r = n(n-1)/2$ puta. Ovi pozivi predstavljaju Ivanova pitanja, u redosljedu kojim ih on postavlja. Vi morate odgovoriti da li postoji direktna avionska linija između različitih gradova u i v. Preciznije, vrijednost koju vraćate treba biti 1 ako postoji direktna avionska linija, odnosno 0 inače.

Podzadaci

Svaki podzadatak sastoji se od nekoliko igara. Osvojićete bodove za neki podzadatak isključivo ako Vaš program pobijedi u svim igrama u korist Andreja.

podzadatak	poeni	n
1	15	$n = 4$
2	27	$4 \leq n \leq 80$
3	58	$4 \leq n \leq 1500$

Implementacioni detalji

Morate poslati tačno jedan fajl sa nazivom `game.c`, `game.cpp` ili `game.pas`. Taj fajl mora implementirati gore opisane funkcije/procedure koristeći sljedeće potpise (signature, zaglavlja).

C/C++ programi

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

Pascal programi

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

Ocjenjivač

Ocjenjivač čita ulaz u sljedećem formatu:

- linija 1: `n`
- narednih r linija: svaka linija sadrži dva cijela broja `u` i `v` koji opisuju pitanje za gradove u and v.