# Hra

Je-Ní-Ček rád hraje hry. Nedávno potkal kamarádku Ma-Řen-Ku a pověděl jí o síti leteckých linek na Tchaj-wanu. Je tam $n$ měst (očíslovaných 0, ..., $n-1$), mezi některými z nich vedou letecké linky. Každá linka spojuje dvě města a může být použita v obou směrech.

Ma-Řen-Ka se zeptala Je-Ní-Čka, zda je možné cestovat mezi každými dvěma městy těmito leteckými linkami (buď přímo, nebo nepřímo). Je-Ní-Ček nechtěl prozradit správnou odpověď a navrhnul následující hru. Ma-Řen-Ka mu může pokládat otázky tvaru „Jsou města $x$ a $y$ *přímo* spojena linkou?“ a Je-Ní-Ček na otázku hned odpoví. Ma-Řen-Ka se postupně zeptá na všechny dvojice měst, na každou právě jednou, dohromady tedy položí $r = n(n-1)/2$ otázek. Ma-Řen-Ka hru vyhraje, pokud existuje takové $i$, že $i < r$ a zároveň po získání odpovědí na prvních $i$ otázek může rozhodnout, zda je možné cestovat mezi každými dvěma městy. V opačném případě, pokud potřebuje všech $r$ odpovědí, vyhraje Je-Ní-Ček.

Aby hra byla pro Je-Ní-Čka zajímavější, dohodli se, že zapomenou na stávající tchajwanskou leteckou síť, Je-Ní-Ček si bude v průběhu hry vymýšlet vlastní síť a volit své odpovědi na základě předchozích otázek. Pomozte Je-Ní-Čkovi odpovídat na otázky tak, aby ve hře zvítězil.

## Příklady

Pravidla hry objasníme na třech příkladech. Ve všech příkladech je $n = 4$ měst a $r = 6$ otázek.

V prvním příkladu (viz následující tabulka) Je-Ní-Ček *prohraje*, protože po otázce s číslem 4 už Ma-Řen-Ka s jistotou ví, že je možné cestovat mezi každou dvojicí měst nezávisle na tom, jaké budou odpovědi na zbývající otázky.

| číslo otázky | otázka | odpověď |
|---|---|---|
| 1 | 0, 1 | ano |
| 2 | 3, 0 | ano |
| 3 | 1, 2 | ne |
| 4 | 0, 2 | ano |
| ----- | -------- | ------ |
| 5 | 3, 1 | ne |
| 6 | 2, 3 | ne |

V dalším příkladu Ma-Řen-Ka po otázce číslo 3 ví, že *není možné* leteckými linkami cestovat z města 0 do města 1 a Je-Ní-Ček proto opět *prohrává*.

| číslo otázky | otázka | odpověď |
|---|---|---|
| 1 | 0, 3 | ne |

| číslo otázky | otázka | odpověď |
|---|---|---|
| 2 | 2, 0 | ne |
| 3 | 0, 1 | ne |
| ----- | -------- | ------ |
| 4 | 1, 2 | ano |
| 5 | 1, 3 | ano |
| 6 | 2, 3 | ano |

V posledním příkladu Ma-Řen-Ka nemůže až do získání odpovědi na poslední otázku rozhodnout, zda je možné cestovat mezi každými dvěma městy, Je-Ní-Ček proto vyhraje. Protože odpověděl *ano* na poslední otázku (v následující tabulce), je možné cestovat mezi každými dvěma městy. Kdyby ovšem odpověděl na poslední otázku *ne* , nebylo by to možné.

| číslo otázky | otázka | odpověď |
|---|---|---|
| 1 | 0, 3 | ne |
| 2 | 1, 0 | ano |
| 3 | 0, 2 | ne |
| 4 | 3, 1 | ano |
| 5 | 1, 2 | ne |
| 6 | 2, 3 | ano |

# Úloha

Napište program, který pomůže Je-Ní-Čkovi vyhrát hru. Uvědomte si, že ani Ma-Řen-Ka, ani Je-Ní-Ček předem neznají strategii toho druhého. Ma-Řen-Ka může pokládat své otázky v libovolném pořadí a Je-Ní-Ček na ně musí okamžitě odpovídat bez znalosti budoucích otázek. Implementujte následující dvě funkce.

- `initialize(n)` -- Na začátku výpočtu bude zavolána funkce `initialize`. Její parametr $n$ udává počet měst.
- `hasEdge(u, v)` -- Potom bude r-krát zavolána funkce `hasEdge`, kde $r = n(n-1)/2$. Každé volání reprezentuje jednu otázku Ma-Řen-Ky v pořadí, v jakém je pokládá. Pro každou otázku musíte rozhodnout, zda vede přímá linka z města $u$ do města $v$. Návratová hodnota je rovna 1, pokud přímá linka z $u$ do $v$ existuje, 0 v opačném případě.

# Podúlohy

Každá podúloha se skládá z několika her. Body za podúlohu získáte pouze tehdy, když váš program vyhraje za Je-Ní-Čka všechny tyto hry.

| podúloha | počet bodů | $n$ |
|---|---|---|
| 1 | 15 | $n = 4$ |
| 2 | 27 | $4 \leq n \leq 80$ |
| 3 | 58 | $4 \leq n \leq 1500$ |

# Upřesnění implementace

Odevzdejte právě jeden soubor pojmenovaný `game.c`, `game.cpp` nebo `game.pas`. Tento soubor implementuje funkce popsané výše s následujícími parametry.

## Programy v C/C++

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

## Programy v Pascalu

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

## Ukázkový vyhodnocovač

Ukázkový vyhodnocovač čte vstup v následujícím formátu:

- řádek 1: `n`
- následujících $r$ řádků: každý řádek obsahuje dvě celá čísla `u` a `v`, která určují dotaz na města $u$ a $v$.