SocialEngineering

Problem Name	Social Engineering
Input file	Interactive task
Output file	Interactive task
Time limit	5 seconds
Memory limit	256 megabytes

Sociálna sieť je neorientovaný graf, ktorý má n vrcholov a m hrán. Vrcholy predstavujú ľudí a hrany sú priateľstvá medzi nimi.

Mária má účet v jednej takejto sociálnej sieti. Rada vypúšťa do sociálnej siete rôzne výzvy. Každá takáto výzva vyzerá tak, že najskôr Mária niečo jednoduché spraví a potom vyzve nejakého svojho priateľa, nech spraví to isté ako ona. Takáto výzva potom cestuje po sociálnej sieti: Máriin priateľ vyzve jedného svojho priateľa, ten zase niekoho zo svojich, a tak ďalej.

Môže sa stať, že konkrétny človek dostane tú istú výzvu postupne viackrát, je však zaručené, že každá *neusporiadaná* dvojica ľudí sa vo výzve vyskytne najviac raz. (Akonáhle človek A vyzve človeka B, už nemôže ani B vyzvať A, ani A znovu vyzvať B.)

Inými slovami, každej výzve zodpovedá nejaká *prechádzka* (anglicky: walk) v našom grafe - teda taká postupnosť po sebe idúcich hrán, v ktorej sa žiadna hrana neopakuje.

Hovoríme, že človek prehral, ak v nejakej výzve príde na rad, ale už nevie ďalej vyzvať žiadneho zo svojich priateľov.

Všetkých $n - 1$ ľudí okrem Márie sa teraz dohodlo, že sa najbližšiu výzvu idú hrať ako hru "všetci proti Márii" - všetci budú spolupracovať a pokúsia sa spolu dosiahnuť, aby Mária prehrala.

Tvojou úlohou je koordinovať všetkých ľudí okrem Márie.

Implementation

Implementuj funkciu

```
void SocialEngineering(int n, int m, vector<pair<int,int>> edges);
```

Túto tvoju funkciu grader práve raz zavolá. Ako parametre dostaneš počet vrcholov (*n*), počet hrán (*m*) a zoznam hrán. Pole so zoznamom hrán bude obsahovať presne *m* prvkov, každý z nich je `pair` obsahujúci čísla dvoch vrcholov spojených hranou. Vrcholy sú očíslované od 1 po *n*, pričom Mária má číslo 1.

Ak na začiatku celej hry platí, že Mária má vyhrávajúcu stratégiu, tvoj program to musí spoznať a odmietnuť hrať. V takomto prípade musíš funkciu `SocialEngineering` ukončiť skôr, ako zavoláš nejakú z funkcií gradera (viď nižšie). Ak tak spravíš, dostaneš na nevyhrateľných vstupoch `Accepted`.

Pre všetky ostatné vstupy musí tvoja funkcia hrať a vyhrať hru proti Márii.

Tvoja funkcia smie pri tom volať dve funkcie implementované v graderi. Toto je prvá z nich:

```
int GetMove();
```

Vždy, keď je na rade Mária, musíš zavolať túto funkciu. (Prvýkrát tak teda musíš spraviť hneď na začiatku hry.)

Ako návratovú hodnotu ti táto funkcia vráti jednu z nasledovných možností:

- Ak Mária vyzve svojho priateľa, ktorý má číslo *v* ($2 \leq v \leq n$), návratovou hodnotou bude číslo *v*. (Ak dostaneš návratovú hodnotu z tohto rozsahu, vždy bude predstavovať platný ťah.)
- Ak sa Mária rozhodne hru vzdať (zväčša preto, že už nemá koho vyzvať - viď sekciu Subtasks), táto funkcia vráti hodnotu 0. Keď dostaneš hodnotu 0, vyhrala si. Ukonči beh tvojej funkcie `SocialEngineering` (sprav return) a dostaneš za tento test verdikt `Accepted`.

Ak túto funkciu zavoláš, keď Mária nie je na ťahu, dostaneš verdikt `Wrong Answer`.

Druhá funkcia gradera, ktorú tiež smieš volať, je táto:

```
void MakeMove(int v);
```

Túto funkciu máš zavolať vždy, keď je na ťahu niekto iný ako Mária. Ako parameter *v* použi číslo toho človeka, ktorého má aktuálny hráč vyzvať.

Ak zavoláš túto funkciu, keď je na ťahu Mária, dostaneš `Wrong Answer`.

Ak zavoláš túto funkciu ale "aktuálny človek vyzve človeka *v*" nie je platný ťah, tiež dostaneš `Wrong Answer`.

Constraints

- $2 \leq n \leq 2 \cdot 10^5$.

- $1 \le m \le 4 \cdot 10^5$.
- Graf je súvislý.
- Každá hrana spája dva rôzne vrcholy.
- Všetky neusporiadané dvojice vrcholov v poli `edges` sú navzájom rôzne.

Subtasks

Mária hrá optimálne: kedykoľvek, keď má víťaznú stratégiu, hrá podľa nej.

Navyše platí, že ak je v prehrávajúcej pozícii, bude sa rôznymi spôsobmi snažiť dostať tvoj program do situácie, v ktorej môže spraviť chybu. S výnimkou podúlohy 3 (viď nižšie) sa Mária vzdá len vtedy, keď už nevie spraviť platný ťah.

V jednotlivých podúlohách platia nasledovné dodatočné obmedzenia a výnimky:

1. (15 points) $n, m \le 10$.
2. (15 points) Každý okrem Márie má nanajvýš dvoch priateľov.
3. (20 points) Ak Mária nemá na začiatku hry zaručene vyhrávajúcu stratégiu, okamžite sa vzdá.
4. (25 points) $n, m \le 100$.
5. (25 points) Bez dodatočných obmedzení.

Sample Interaction

Tvoj program spravil	Grader spravil	Vysvetlenie
-	`SocialEngineering(5, 6, {{1,4}, {1,5}, {2,4}, {2,5}, {2,3}, {3,5}})`	Grader zavolal tvoju funkciu `SocialEngineering` a odovzdal jej graf s 5 vrcholmi a 6 hranami.
`GetMove()`	`return 4`	Grader ťa informuje, že Mária vyzvala človeka 4.
`MakeMove(2)`	-	Tvoj program oznámil graderu, že človek 4 vyzval človeka 2.
`MakeMove(5)`	-	Tvoj program oznámil graderu, že človek 2 vyzval človeka 5.
`MakeMove(1)`	-	Tvoj program oznámil graderu, že človek 5 vyzval človeka 1 (teda Máriu).

Tvoj program spravil	Grader spravil	Vysvetlenie
`GetMove()`	`return 0`	Mária nemá koho vyzvať, takže sa vzdá.
`return`	-	Ukončila si beh funkcie `SocialEngineering` a vyhrala si.

Tvoj program spravil	Grader spravil	Vysvetlenie
-	`SocialEngineering(2, 1, {{1,2}})`	Grader zavolal tvoju funkciu `SocialEngineering` a odovzdal jej graf s 2 vrcholmi a 1 hranou.
`return`	-	Mária má pre tento graf vyhrávajúcu stratégiu. Tvoj program to musí spoznať a spraviť return z tvojej funkcie bez toho, aby volal `GetMove`.

Sample Grader

V balíčku `SocialEngineering.zip` dostaneš ukážkový grader v súbore `grader.cpp`. Tento grader očakáva vstup v nasledovnom formáte:

- V prvom riadku sú čísla *n* a *m* (najskôr počet vrcholov, potom hrán).
- V každom z nasledujúcich *m* riadkov sú dve čísla vrcholov spojených hranou.

Ukážkový grader načíta vstup a potom zavolá tvoju funkciu `SocialEngineering` a odovzdá jej načítaný graf.

Upozorňujeme, že tento ukážkový grader neimplementuje Máriinu vyhrávajúcu stratégiu, slúži len na ukážku interakcie.

Skompilovať dokopy grader a svoje riešenie vieš nasledovným príkazom v termináli:

```
g++ -std=gnu++11 -O2 -o solution grader.cpp solution.cpp
```

Vo vyššie uvedenom príkaze `solution.cpp` je názov súboru s tvojim riešením (t.j. toho, čo odovzdávaš do CMS).

Takto skompilovaný program vieš spustiť na príklade vstupu (tie sú tiež v balíčku) nasledovným príkazom:

```
./solution < input.txt
```