Zadanie: SKO
Skorpion

Olimpiada Informatyczna

ONTAK 2013, dzień 6. Plik źródłowy `sko.*` Dostępna pamięć: 256 MB. *13.08.2013*

Są takie miejsca na świecie, w których wszystko usiłuje człowieka zabić. Nawet grafy nieskierowane.

Graf nieskierowany jest *skorpionem*, jeśli posiada trzy charakterystyczne wierzchołki:

- *żądło*, które jest połączone krawędzią tylko z ogonem,
- *ogon*, który jest połączony krawędzią tylko z żądłem i odwłokiem,
- *odwłok*, połączony krawędziami ze wszystkimi wierzchołkami grafu z wyjątkiem żądła.

Pozostałe wierzchołki mogą być dowolnie połączone. Masz dany graf o n wierzchołkach, który z daleka wygląda niewinnie — to dlatego, że nie widzisz zbyt dobrze jego krawędzi. Możesz wykonać pewną liczbę zapytań postaci *„czy między wierzchołkami x i y istnieje krawędź?”*. Na podstawie odpowiedzi na nie rozstrzygnij, czy graf jest skorpionem.

Komunikacja programu

Twój program nie powinien nic czytać z wejścia ani wypisywać na wyjście. Zamiast tego zostanie on skompilowany z biblioteką oceniającą, która będzie odpowiadać na zapytania o krawędzie. Na początku programu umieść dyrektywę:

```
#include "sko.h"
```

Biblioteka udostępnia trzy funkcje:

- `int init()` — funkcję tę Twój program powinien wykonać raz, na początku. Zwraca ona n — liczbę wierzchołków ($8 \leq n \leq 20\,000$). Wierzchołki są ponumerowane od 1 do n.
- `bool edge(int x, int y)` — zapytanie o krawędź (nieskierowaną) (x, y). Zwraca **true** wtedy i tylko wtedy, gdy krawędź występuje w grafie.
- `void answer(bool skorpion, int zadlo, int ogon, int odwlok)` — ta funkcja powinna zostać wywołana raz, na końcu działania programu. Informuje ona bibliotekę o Twojej odpowiedzi. Pierwszy parametr (`skorpion`) powinien być równy **true** wtedy i tylko wtedy, gdy graf jest skorpionem. W tym przypadku parametry `zadlo`, `ogon` i `odwlok` powinny oznaczać numery wierzchołków będących odpowiednio żądłem, ogonem i odwłokiem skorpiona. Jeśli graf nie jest skorpionem, parametry te nie mają znaczenia.

Ocenianie

Za każdy test otrzymasz punkty, jeśli Twój program udzieli prawidłowej odpowiedzi i zmieści się w odpowiednim limicie zapytań:

- Za wykonanie nie więcej niż $6n$ zapytań otrzymasz 100% punktów za daną grupę.
- Za wykonanie $10n$ zapytań otrzymasz 90% punktów za daną grupę.
- Za wykonanie $30n$ zapytań otrzymasz 60% punktów za daną grupę.
- Za wykonanie $100n$ zapytań otrzymasz 30% punktów za daną grupę.

Pomiędzy podanymi granicami $(6n, 10n, 30n, 100n)$ punktacja spada liniowo. Przekroczenie $100n$ zapytań powoduje automatycznie 0 punktów za test. Testy łączone są w grupy – punktacja za grupę testów to minimum z punktacji poszczególnych testów.

Dodatkowe ograniczenia

- W testach wartych 14% punktów pozostałe wierzchołki (poza żądłem, ogonem i odwłokiem) stanowią klikę (każda para jest połączona).
- W testach wartych 14% punktów liczba krawędzi nie przekracza $4n$.

Uwaga: W niektórych testach biblioteka oceniająca tworzy graf w trakcie działania programu użytkownika, ustalając stan krawędzi dopiero przy pierwszym zapytaniu o nią.

Przykładowa biblioteka

Do Twojej dyspozycji jest przykładowa biblioteka `skozaw.cpp`, która implementuje podane trzy funkcje: `init()` wczytuje opis grafu ze standardowego wejścia, `edge()` i `answer()` sprawdzają wczytany graf. Na wejściu w pierwszym wierszu powinna być liczba wierzchołków n, w kolejnym – numery żądła, ogona i odwłoka, zaś w n następnych wierszach macierz sąsiedztwa grafu. Przykładowy graf znajduje się w pliku `skozaw.in`. Aby przetestować na nim swój program, wpisz komendę:

```
g++ program.cpp skozaw.cpp -oskorpion -O2
```

a następnie

```
./skorpion < skozaw.in
```