Zadanie: KOL
Kółko informatyczne [B]

POTYCZKI ALGORYTMICZNE

Potyczki Algorytmiczne 2014, runda 4. Dostępna pamięć: 128 MB. *15.05.2014*

Uwaga! To jest zadanie *rozproszone*. Zanim zaczniesz je rozwiązywać, zapoznaj się z informacjami dotyczącymi tego typu zadań dostępnymi w serwisie internetowym zawodów w zakładce *Zadania rozproszone*.

Na kółku informatycznym uczniowie siedzą... w kółku... i rozwiązują zadania. Gdy ktoś wymyśli rozwiązanie, natychmiast dzieli się nim ze swoimi obydwoma sąsiadami (co trwa minutę), po czym oni przekazują je dalej (co również trwa minutę)... i tak w kółko.

Jeśli kwadrans po dwunastej Janek wpadnie na genialny pomysł, o której godzinie usłyszy o nim Krzyś? Ilu minut potrzeba, żeby rozwiązanie dotarło od Kasi do Tomka? Na takie pytania będzie musiał odpowiedzieć Twój program.

Dane wejściowe

Twój program nie może używać standardowego wejścia. Zamiast tego dane testowe zostaną mu udostępnione za pośrednictwem dostarczonej biblioteki interaktywnej. Aby jej użyć, wpisz w swoim programie:

- C/C++: `#include "kollib.h"`
- Pascal: `uses kollib;`

Biblioteka udostępnia sześć funkcji:

- `NumberOfStudents` – zwraca liczbę uczniów n uczestniczących w kółku ($3 \leq n \leq 10^9$). Uczniowie są numerowani kolejnymi liczbami naturalnymi od 1 do n.
  - C/C++: `int NumberOfStudents();`
  - Pascal: `function NumberOfStudents(): longint;`
- `FirstNeighbor` – dla i-tego ucznia ($1 \leq i \leq n$) zwraca numer jego pierwszego sąsiada. Uczniowie miewają problemy z odróżnieniem strony lewej od prawej, więc na wszelki wypadek wolą mówić o swoich sąsiadach w kolejności ich numerów. Innymi słowy, „pierwszy sąsiad" danego ucznia będzie miał zawsze mniejszy numer niż jego „drugi sąsiad".
  - C/C++: `int FirstNeighbor(int i);`
  - Pascal: `function FirstNeighbor(i: longint): longint;`
- `SecondNeighbor` – dla i-tego ucznia ($1 \leq i \leq n$) zwraca numer jego drugiego sąsiada.
  - C/C++: `int SecondNeighbor(int i);`
  - Pascal: `function SecondNeighbor(i: longint): longint;`
- `NumberOfQueries` – zwraca liczbę zapytań m ($0 \leq m \leq 200$), na które powinien odpowiedzieć Twój program. Zapytania są numerowane kolejnymi liczbami naturalnymi od 1 do m.
  - C/C++: `int NumberOfQueries();`
  - Pascal: `function NumberOfQueries(): longint;`
- `QueryFrom` – dla i-tego zapytania ($1 \leq i \leq m$) zwraca numer ucznia, który wymyślił rozwiązanie.
  - C/C++: `int QueryFrom(int i);`
  - Pascal: `function QueryFrom(i: longint): longint;`
- `QueryTo` – dla i-tego zapytania ($1 \leq i \leq m$) zwraca numer ucznia, o którym chcemy wiedzieć, po ilu minutach pozna rozwiązanie.
  - C/C++: `int QueryTo(int i);`
  - Pascal: `function QueryTo(i: longint): longint;`

W dziale *Pliki* w systemie SIO2 znajduje się archiwum zawierające przykładowe pliki bibliotek oraz (niepoprawne) rozwiązania ilustrujące sposób ich użycia.

Wyjście

Twój program powinien wypisać na wyjście dokładnie m wierszy. W i-tym wierszu wyjścia powinna znaleźć się odpowiedź na i-te zapytanie, czyli liczba minut potrzebnych na przekazanie rozwiązania pomiędzy uczniami.

Komunikacja

Podczas oceny Twojego programu system sprawdzający uruchomi jednocześnie wiele jego instancji, każdą na osobnym komputerze. Instancje powinny komunikować się za pomocą biblioteki `message`. W tym celu w programie należy umieścić wiersz:

- C/C++: `#include "message.h"`
- Pascal: `uses message;`

Instrukcja dotycząca używania tej biblioteki jest dostępna w serwisie internetowym zawodów w zakładce *Zadania rozproszone*.

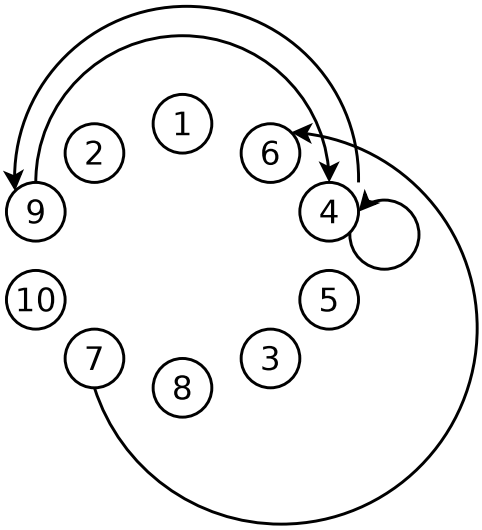
Przykładowy przebieg programu

Dla przebiegu programu:

Wywołanie funkcji	**Zwrócona wartość**
`NumberOfStudents();`	10
`FirstNeighbor(1)`	2
`SecondNeighbor(1);`	6
`FirstNeighbor(2);`	1
`SecondNeighbor(2);`	9
`FirstNeighbor(3);`	5
`SecondNeighbor(3);`	8
...	...
`FirstNeighbor(10);`	7
`SecondNeighbor(10);`	9
`NumberOfQueries();`	4
`QueryFrom(1);`	7
`QueryTo(1);`	6
`QueryFrom(2);`	4
`QueryTo(2);`	9
`QueryFrom(3);`	4
`QueryTo(3);`	4
`QueryFrom(4);`	9
`QueryTo(4);`	4

poprawnym wynikiem jest:

```
5
4
0
4
```



Testy przykładowe

Po wysłaniu rozwiązania od razu poznasz wyniki Twojego programu na poniższych testach:

- `0a`: test przykładowy z treści zadania, uruchamiany na 10 komputerach;
- `0b`: $n = 200\,000\,000$, najpierw siedzą kolejno uczniowie o numerach nieparzystych, po nich siedzą kolejno uczniowie o numerach parzystych; $m = 200$, w zapytaniach uczniowie losowani niezależnie z jednostajnym prawdopodobieństwem; test jest uruchamiany na 20 komputerach.