Zadanie: OFE
Oferty

Olimpiada Informatyczna

ONTAK 2015, dzień czwarty. Plik źródłowy `ofe.*` Dostępna pamięć: 256 MB. *13.7.2015*

Do niedawna w Bajtocji obowiązywał przepis mówiący że publiczne przetargi wygrywa zawsze ten podmiot, który zaoferuje najniższą cenę. Nieraz prowadziło to do problemów. Kiedyś, na przykład, informatyczny system obsługi wyborów nowego błazna Bajtocji* realizowała firma, która co prawda nie miała pojęcia, jak się za to zabrać, ale za to zaproponowała najniższą cenę.

Król postanowił to w końcu zmienić i wprowadził prawo, na mocy którego przetargi będzie wygrywała druga w kolejności niemalejącej oferta. Owszem, to rozwiązanie też ma dużo wad, ale kraj trzeba zmieniać małymi krokami.

Tobie, jako nadwornemu informatykowi, przypadło zadanie obsługi ofert. Musisz po prostu napisać program, który znajdzie drugą najtańszą ofertę. Brzmi prosto? Mamy nadzieję, że umiesz zrobić to szybko i sprawnie, bo w przeciwnym wypadku czeka Cię spotkanie z królewskim katem, któremu dopiero co zaczęła się kadencja.

Ach... prawie zapomnieliśmy powiedzieć, że ze względów bezpieczeństwa i w ramach walki z korupcją, nie możesz poznać wartości ofert. Możesz jedynie porównywać dwie oferty ze sobą i sprawdzać, która z nich jest tańsza. Bądź świadom, że im więcej porównań wykonasz, tym większa jest szansa, że kat zdąży wrócić z urlopu...

Komunikacja programu

Twój program nie powinien nic czytać ze standardowego wejścia ani nic wypisywać na standardowe wyjście. Zamiast tego zostanie on skompilowany z królewską biblioteką, przy pomocy której będzie mógł porównywać oferty. Na początku swojego programu umieść dyrektywę:

```
#include "ofe.h"
```

Biblioteka udostępnia trzy funkcje:

- `int getN()` — zwraca liczbę ofert n ($2 \leq n \leq 10\,000$), które musisz rozważyć.
- `bool cmp(int a, int b)` — zwraca `true`, jeśli oferta o numerze a jest tańsza od oferty od oferty o numerze b, zaś `false` w w przeciwnym przypadku. Oferty numerowane są kolejnymi liczbami od 1 do n. Możesz założyć, że żadne dwie oferty nie mają takiej samej ceny.
- `void answer(int ans)` — Twój program powinien wywołać tę funkcję, podając numer drugiej w kolejności rosnącej oferty. Wywołanie funkcji automatycznie zakończy Twój program.

Ocenianie

Liczba punktów, które Twój program dostanie za dany test, zależy od tego, ilu użyje porównań. Oznaczmy przez k liczbę porównań potrzebnych w najgorszym przypadku do znalezienia drugiej najtańszej spośród n ofert.

- Jeżeli Twój program użyje nie więcej niż k porównań, to dostanie 100% punktów za dany test.
- W przeciwnym przypadku, jeżeli użyje nie więcej niż $\lfloor \frac{3}{2}k \rfloor$ porównań, to otrzyma 50%.
- W przeciwnym przypadku, jeżeli użyje nie więcej niż $2k$ porównań, to otrzyma 25% punktów.
- W przeciwnym przypadku Twój program nie otrzyma żadnych punktów za dany test.

Przykład

wywołanie	wynik	komentarz
`getN()`	3	mamy 3 oferty
`cmp(1,2)`	`true`	oferta 1 < oferta 2
`cmp(1,3)`	`true`	oferta 1 < oferta 3
`cmp(2,3)`	`false`	oferta 2 > oferta 3
`answer(3)`		wiemy, że oferta 1 < oferta 3 < oferta 2

*W Bajtocji panuje monarchia, ale po usłyszeniu o demokracji obywatele wywalczyli sobie prawo wyboru błazna, który będzie zabawiał monarchę.

Eksperymenty

W dziale Pliki na SIO znaleźć można przykładową bibliotekę `ofezaw.cpp`. Aby przetestować swój program używając przykładowej biblioteki, skompiluj go poniższą komendą, podstawiając pod `ofe.cpp` nazwę pliku z Twoim rozwiązaniem.

```
g++ -O2 -static -std=c++11 ofe.cpp ofezaw.cpp -o ofe
```

Spowoduje to utworzenie pliku wykonywalnego o nazwie `ofe`, który należy uruchomić. Po uruchomieniu, na standardowego wejście należy podać liczbę ofert n, a następnie n liczb całkowitych – kolejne wartości ofert. Biblioteka będzie odpowiadać na zapytania zgodnie z podanymi wartościami, a następnie wypisze na standardowe wyjście liczbę zapytań oraz sprawdzi, czy faktycznie Twój program podał jako odpowiedź drugą najmniejszą liczbę.