Zadanie: LIC
Licytacja

Olimpiada Informatyczna

XIX OI, etap III, dzień pierwszy. Plik źródłowy `lic.*` Dostępna pamięć: 128 MB. *28.03.2012*

Alojzy i Bajtazar grają w licytację. Do grania w tę grę potrzebny jest bardzo duży zestaw kamyków. Gracze wykonują ruchy na przemian: najpierw Alojzy, potem Bajtazar, znowu Alojzy itd. W danym momencie gry istotne są dwie wartości: aktualna stawka i aktualny rozmiar stosu. Gra zaczyna się od stawki jednego kamyka i pustego stosu. W każdym ruchu gracz wykonuje jeden z następujących ruchów:

- podwaja stawkę,
- potraja stawkę,
- pasuje.

W przypadku, gdy gracz spasuje, cała aktualna stawka wędruje na stos (jest to jedyny sposób powiększenia stosu), a licytacja ponownie zaczyna się od stawki jednego kamyka. **Jeżeli gracz spasuje, to następny ruch należy do jego przeciwnika (Alojzy zaczyna tylko całą rozgrywkę).** Gracz, który spowoduje przepełnienie stosu (następuje to, gdy na stosie znajduje się n lub więcej kamyków), przegrywa. Jeśli przed ruchem gracza łączna liczba kamyków na stosie i w stawce osiąga lub przekracza n, gracz ten nie może już podwoić ani potroić stawki. Musi spasować, tym samym dokładając stawkę na stos, co powoduje jego przegraną.

Alojzy bardzo często przegrywa. Bajtazar zaproponował mu ciekawe wyzwanie — zamiast samemu grać w licytację, lepiej napisać programy, które będą w nią grały. Niestety, Alojzy nie umie programować. Pomóż mu!

Napisz program, który będzie grał w licytację w imieniu Alojzego przeciw bibliotece napisanej przez Bajtazara.

Ocenianie

We wszystkich przypadkach testowych Twój program będzie mógł wygrać (o ile wykona odpowiednie ruchy) niezależnie od ruchów biblioteki. Twój program otrzyma punkty za dany test tylko wtedy, gdy wygra z biblioteką.

We wszystkich testach zachodzi warunek $1 \leqslant n \leqslant 30\,000$. W 50% przypadków testowych zachodzi dodatkowy warunek $n \leqslant 25$.

Opis użycia biblioteki

Aby użyć biblioteki, należy wpisać na początku programu:

- **C/C++:** `#include "cliclib.h"`
- **Pascal:** `uses pliclib;`

Biblioteka udostępnia następujące funkcje i procedury:

- `inicjuj` — zwraca liczbę n. Powinna zostać użyta dokładnie raz, na samym początku działania programu.
  - **C/C++:** `int inicjuj();`
  - **Pascal:** `function inicjuj: longint;`
- `alojzy` — informuje bibliotekę o ruchu Twojego programu. Jej jedynym parametrem jest liczba całkowita x, która oznacza wykonany ruch: $x = 1$ oznacza spasowanie, $x = 2$ — podwojenie stawki, natomiast $x = 3$ — potrojenie stawki.
  - **C/C++:** `void alojzy(int x);`
  - **Pascal:** `procedure alojzy(x: longint);`
- `bajtazar` — funkcja informuje Twój program o ruchu biblioteki. Zwraca jedną liczbę x, która oznacza wykonany ruch. Analogicznie jak w przypadku funkcji `alojzy`, $x = 1$ oznacza pas, $x = 2$ — podwojenie, natomiast $x = 3$ — potrojenie stawki.
  - **C/C++:** `int bajtazar();`
  - **Pascal:** `function bajtazar: longint;`

Po wywołaniu funkcji `inicjuj` należy naprzemiennie wywoływać funkcje `alojzy` oraz `bajtazar` (w takiej kolejności). Złamanie protokołu komunikacji zostanie potraktowane jako błędna odpowiedź i spowoduje przyznanie 0 punktów za dany test. W tym zadaniu użycie standardowego wejścia i wyjścia jest **zabronione**. Jakakolwiek komunikacja powinna odbywać się tylko za pośrednictwem powyżej podanych funkcji i procedur. Biblioteka zakończy działanie programu automatycznie po zakończeniu gry.

Rozwiązanie będzie kompilowane wraz z biblioteką przy użyciu następujących poleceń:

- **C:** `gcc -O2 -static cliclib.c lic.c -lm`
- **C++:** `g++ -O2 -static cliclib.c lic.cpp -lm`
- **Pascal:** `ppc386 -O2 -Xs -Xt lic.pas`

Eksperymenty

W katalogu **/home/zawodnik/rozw/lic/** znajdują się przykładowe pliki bibliotek i przykładowe nieoptymalne rozwiązania ilustrujące sposób ich użycia (można je także pobrać w dziale *Przydatne zasoby* w SIO). Program skompilowany z przykładową biblioteką wczytuje ze standardowego wejścia liczbę n, a następnie aż do zakończenia gry wczytuje kolejne ruchy wykonywane przez Bajtazara, symulując zaimplementowaną strategię Alojzego. Na standardowe wyjście diagnostyczne (`stderr`) program wypisuje szczegółowy przebieg gry. **W testach ocen, zarówno na komputerze zawodnika, jak i w SIO, Bajtazar odpowiada kolejno: pas, podwojenie stawki, potrojenie stawki, pas, podwojenie stawki, potrojenie stawki,...**

Ostateczna ocena programów odbędzie się z wykorzystaniem innego zestawu bibliotek.

W przypadku tego zadania w SIO nie jest dostępna opcja *Test programu*.

Przykładowy przebieg programu

C/C++	Pascal	Wynik	Stos	Stawka	Wyjaśnienie
`n = inicjuj();`	`n := inicjuj;`	15	0	1	Od tego momentu $n = 15$.
`alojzy(2);`	`alojzy(2);`	—	0	2	Twój program podwoił stawkę.
`bajtazar();`	`bajtazar;`	2	0	4	Biblioteka odpowiedziała podwojeniem stawki.
`alojzy(3);`	`alojzy(3);`	—	0	12	Twój program potroił stawkę.
`bajtazar();`	`bajtazar;`	1	12	1	Biblioteka spasowała. 12 kamyków wędruje na stos, a w stawce ponownie jest tylko 1 kamyk.
`alojzy(2);`	`alojzy(2);`	—	12	2	Twój program podwoił stawkę.
`bajtazar();`	`bajtazar;`	3	12	6	Biblioteka potroiła stawkę.
`alojzy(1);`	`alojzy(1);`	—	18	1	Łączna liczba kamyków na stosie i w stawce przekroczyła 15. Twój program jest zmuszony spasować.

Powyższy przebieg programu jest poprawny, ale nieoptymalny. Twój program nie dostałby punktów za ten test. W szczególności, dla $n = 15$ istnieje możliwość wygranej Alojzego, niezależnie od ruchów Bajtazara.