Zadanie: WAG
Waga

Olimpiada Informatyczna

XXIX OI, etap III, dzień drugi. Plik źródłowy `wag.*` Dostępna pamięć: 256 MB. *7.04.2022*

Bajtazar kupił właśnie wagę szalkową z zestawem siedmiu odważników o masach 1, 2, 3, 4, 5, 6 oraz 7 kilogramów. Niestety, odważniki nie różnią się od siebie wyglądem, a na żadnym z nich nie widnieje żaden nadruk. Do zestawu dołączono naklejki z liczbami 1, 2, 3, 4, 5, 6 oraz 7, ale nie dołączono instrukcji, na który odważnik przykleić którą naklejkę. Na razie więc Bajtazar przykleił naklejki wedle swojego uznania (po jednej naklejce na jeden odważnik), ale docelowo chciałby, aby każda naklejka symbolizowała wagę odważnika.

Bajtazar wpadł na genialny pomysł, jak to zrobić. Może przecież kłaść na szalkach wagi niektóre odważniki i patrzeć, w którą stronę waga się przechyla. Pomożesz Bajtazarowi ustalić właściwe przypisanie naklejek do odważników przy użyciu małej liczby ważeń?

Komunikacja

Należy zaimplementować program, który rozwiąże problem Bajtazara, korzystając z dostarczonej biblioteki (symulującej wagę). Aby użyć biblioteki, należy wpisać w swoim programie:

- C/C++: `#include "waglib.h"`
- Python: `from waglib import liczba_testow, poloz_lewo, poloz_prawo, odloz, wazenie, odpowiedz`

Biblioteka udostępnia następujące funkcje:

- `liczba_testow`() – Podczas jednego uruchomienia programu będzie trzeba wielokrotnie pomóc Bajtazarowi (dla różnych zestawów odważników). Ta funkcja zwraca liczbę całkowitą t, oznaczającą liczbę przypadków testowych do rozważenia w danym uruchomieniu. Możesz założyć, że w teście przykładowym $t = 2$, a w pozostałych testach $t = 5040$.
- `poloz_lewo`(x) – Powoduje położenie odważnika z naklejką x ($1 \leq x \leq 7$) na lewej szalce wagi.
- `poloz_prawo`(x) – Powoduje położenie odważnika z naklejką x ($1 \leq x \leq 7$) na prawej szalce wagi.
- `odloz`(x) – Powoduje odłożenie odważnika z naklejką x ($1 \leq x \leq 7$) na stół (tzn. zdjęcie z szalek wagi). Funkcje `poloz_lewo`, `poloz_prawo` oraz `odloz` można wywołać niezależnie od położenia odważnika x przed wywołaniem. Jeżeli odważnik już znajdował się w miejscu, gdzie próbujesz go ustawić, to wywołanie funkcji nie ma żadnego efektu.
- `wazenie`() – Zwraca liczbę ze zbioru $\{-1, 0, 1\}$ oznaczającą odpowiednio: lewa szalka cięższa, waga w równowadze, prawa szalka cięższa.
- `odpowiedz`(T) – Ta funkcja pozwala nam zgłosić rozwiązanie problemu Bajtazara. Należy ją wykonać dokładnie raz dla każdego przypadku testowego.

  Przyjmuje ona tablicę $T[0..6]$ (daną jako wektor `std::vector<int>` w języku C/C++ lub lista w języku Python), która oznacza, że odważnik, na którym Bajtazar tymczasowo przykleił naklejkę z liczbą i, waży $T[i-1]$.

  Po wywołaniu funkcji `odpowiedz` program automatycznie przechodzi do kolejnego przypadku testowego (w szczególności wszystkie odważniki, które były na wadze, zostają z niej zdjęte). Po wykonaniu tej funkcji na ostatnim teście program automatycznie zostanie zakończony.

Biblioteka **nie musi** przydzielać naklejek do odważników na początku interakcji z Twoim programem. Może ona w trakcie interakcji zmieniać wagi odważników, o ile nowe wagi są nadal spójne z wynikami zwróconymi przez dotychczasowe wywołania funkcji `wazenie`.

Twój program nie może otwierać żadnych plików, ani używać standardowego wejścia i wyjścia. Rozwiązanie będzie kompilowane wraz z biblioteką następującymi poleceniami:

- C++: `g++ -O3 -static waglib.cpp wag.cpp`
- Python: `python3 wag.py`

Ocenianie

Twój program będzie testowany na jednej grupie testów.

Jeśli R to maksymalna liczba wywołań funkcji `wazenie`, które Twój program wykonał w jednym przypadku testowym, to Twoje rozwiązanie otrzyma następującą liczbę punktów (odpowiednio przeskalowaną w przypadku przekroczenia połowy limitu czasowego):

Olimpiada Informatyczna finansowana jest ze środków Ministerstwa Edukacji Narodowej w ramach zadania publicznego „Organizacja i przeprowadzenie olimpiad i turniejów w latach szkolnych 2019/2020, 2020/2021, 2021/2022”.

Liczba wywołań	Liczba punktów
$R \leq 9$	100 punktów za test
$10 \leq R \leq 15$	$40 + 10(15 - R)$ punktów za test
$16 \leq R \leq 25$	$10 + 3(25 - R)$ punktów za test
$R \geq 26$	0 punktów za test (werdykt *Błędna odpowiedź*)

Przykładowy przebieg programu

Poniżej przedstawiono przebieg programu dla testu przykładowego, w którym mamy $t = 2$ przypadki testowe. Niech w_i oznacza wagę odważnika, na który Bajtazar przykleił naklejkę z liczbą i.

W pierwszym przypadku naklejki są przyklejone prawidłowo ($w_i = i$ dla każdego i) i udaje nam się to zweryfikować za pomocą trzech wywołań funkcji `wazenie`.

W drugim przypadku mamy $w_1 = 2$, $w_2 = 1$, $w_3 = 3$, $w_4 = 7$, $w_5 = 4$, $w_6 = 5$ i $w_7 = 6$. Jeśli w tym przypadku również użyjemy co najwyżej 9 wywołań funkcji `wazenie`, to dostaniemy 100 punktów za ten test.

Wywołanie	Wynik	Opis

`poloz_lewo(1)`
`poloz_lewo(2)`
`poloz_lewo(3)`
`poloz_prawo(7)`
	–	Na lewej szalce 1, 2, 3; na prawej 7

`wazenie()`	1	$w_1 + w_2 + w_3 < w_7$, zatem musi być $\{w_1, w_2, w_3\} = \{1, 2, 3\}$ oraz $w_7 = 7$

`odloz(1)`
`odloz(7)`
`poloz_prawo(4)`
	–	Na lewej szalce 2, 3; na prawej 4

`wazenie()`	-1	$w_2 + w_3 > w_4$ oraz $w_4 \geq 4$, zatem musi być $w_1 = 1$ oraz $w_4 = 4$

`odloz(3)`
`odloz(4)`
`poloz_lewo(4)`
`poloz_prawo(6)`
	–	Na lewej szalce 2, 4; na prawej 6

`wazenie()`	0	$w_2 + 4 = w_6$ oraz $w_2 \in \{2, 3\}$ i $w_6 \in \{5, 6\}$, zatem musi być $w_2 = 2$, $w_6 = 6$, a w konsekwencji $w_3 = 3$ i $w_5 = 5$
`odpowiedz({1,2,3,4,5,6,7})`	–	Prawidłowo odpowiadamy, że $w_i = i$ dla każdego i i przechodzimy do nowego przypadku testowego; szalki są opróżniane

`poloz_lewo(1)`
`poloz_lewo(2)`
`poloz_lewo(3)`
`poloz_prawo(7)`
	–	Na lewej szalce 1, 2, 3; na prawej 7

`wazenie()`	0	$w_1 + w_2 + w_3 = w_7$, zatem musi być $\{w_1, w_2, w_3\} = \{1, 2, 3\}$ oraz $w_7 = 6$ albo $\{w_1, w_2, w_3\} = \{1, 2, 4\}$ oraz $w_7 = 7$
...	...	...
`odpowiedz({2,1,3,7,4,5,6})`	–	Odpowiadamy prawidłowo i program kończy działanie

Eksperymenty

Przykładowe **błędne** rozwiązania wraz z przykładowymi bibliotekami znajdują się w folderze `dlazaw`. Biblioteki mogą różnić się zachowaniem od tych na sprawdzaczkach i nie spełniać założeń zadania. Mają one jedynie pokazać sposób interakcji z programem.