Zadanie: POW
Powtórzenia

Olimpiada Informatyczna

Eliminacje do IOI, dzień próbny. Plik źródłowy `pow.*` Dostępna pamięć: 512 MB. *10.08.2020*

Każdego roku na święta Bożego Narodzenia Bajtazar dekoruje swój dom łańcuchem złożonym z różnokolorowych lampek. W tym roku Bajtazar samemu dobrał kolory lampek, które wchodzą w skład łańcucha, jednak ciężko mu się było zdecydować i ostatecznie łańcuch wyszedł jakiś taki za długi.

Żona Bajtazara przeczytała na pewnym popularnym portalu, że tegoroczna moda świąteczna wymaga, by łańcuch zawierał jak najwięcej różnych powtórzeń. *Powtórzenie* w łańcuchu to ciąg dwóch lub więcej kolejnych lampek tego samego koloru. Powtórzenia nazywamy *różnymi*, jeśli składają się z innego koloru lampek lub mają różne długości. Na przykład ciąg lampek $1, 1, 1, 1, 2, 2, 2, 1, 1, 1$, w którym liczby 1 i 2 oznaczają różne kolory, zawiera pięć różnych powtórzeń:

- $1, 1$
- $1, 1, 1$
- $1, 1, 1, 1$
- $2, 2$
- $2, 2, 2$

Bajtazar zamierza skrócić swój łańcuch poprzez odrzucenie pewnej liczby początkowych i/lub końcowych lampek. Chciałby jednak wiedzieć, ile różnych powtórzeń pozostanie w łańcuchu po takiej operacji. Nasz bohater nie jest jeszcze pewien, ile dokładnie lampek zamierza usunąć, więc przydałaby mu się biblioteka, który pomoże mu obliczać żądane wartości dla różnych możliwych sposobów usunięcia lampek.

Komunikacja

Twoja biblioteka musi udostępniać następujące funkcje:

- `inicjuj(n, t)`
  Ta funkcja zostanie wywołana tylko raz, na początku działania programu. Możesz jej użyć, aby poznać liczbę lampek w łańcuchu n oraz kolory kolejnych lampek, zadane przez wektor t. Kolory lampek są oznaczone liczbami całkowitymi od 0 do 10^9. Numerujemy je tak jak elementy wektora t, liczbami od 0 do $n-1$.
  - **C++:** `void inicjuj(int n, vector<int> t);`
- `ile_powtorzen(a, b)`
  Wywołanie tej funkcji oznacza zapytanie o liczbę różnych powtórzeń we fragmencie łańcucha od a-tej do b-tej lampki włącznie ($0 \le a \le b < n$).
  - **C++:** `int ile_powtorzen(int a, int b);`

Twój program **nie może** czytać żadnych danych (ani ze standardowego wejścia, ani z plików). **Nie może** również nic wypisywać do plików ani na standardowe wyjście. Może pisać na standardowe wyjście diagnostyczne (`stderr`) – pamiętaj jednak, że zużywa to cenny czas. **Nie deklaruj** też funkcji `main`.

Przykładowy przebieg programu

Operacja	Wynik	Wyjaśnienie
`inicjuj(10, [1,1,1,1,2,2,2,1,1,1])`	-	$n = 10$
`ile_powtorzen(0, 9)`	5	zapytanie o cały ciąg
`ile_powtorzen(0, 6)`	5	zapytanie o fragment $[1, 1, 1, 1, 2, 2, 2]$
`ile_powtorzen(4, 6)`	2	zapytanie o fragment $[2, 2, 2]$
`ile_powtorzen(3, 4)`	0	zapytanie o fragment $[1, 2]$

Ocenianie

Zestaw testów dzieli się na następujące podzadania. Testy do każdego podzadania składają się z jednej lub większej liczby osobnych grup testów.

Niech q oznacza łączną liczbę zapytań. We wszystkich podzadaniach zachodzi $1 \le n \le 250\,000, 1 \le q \le 500\,000$.

Podzadanie	Ograniczenia	Punkty
1	$n, q \leq 200$	5
2	$n, q \leq 5000$	10
3	$n, q \leq 20\,000$	10
4	wszystkie fragmenty w zapytaniach `ile_powtorzen` mają tę samą długość	10
5	lampki są tylko dwóch kolorów	10
6	odpowiednio parametry a i b w zapytaniach `ile_powtorzen` są niemalejące	5
7	parametry a w zapytaniach `ile_powtorzen` są niemalejące	10
8	$n, q \leq 100\,000$	15
9	brak dodatkowych ograniczeń	25

Testy „ocen”:

`1ocen:` $n = 10$, $q = 5$, lampki kolorów $1, 1, 2, 2, 3, 3, 4, 4, 5, 5$;

`2ocen:` $n = 5000$, $q = 5000$, lampki kolorów $1, 1, 2, 2, 1, 1, 2, 2, \ldots$; zapytania spełniają wymagania podzadania 7;

`3ocen:` $n = 245\,350$, $q = 500\,000$, jedna lampka koloru 1, dwie koloru 2, trzy koloru 3, ...

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.

Polecenie kompilacji wygląda następująco, przy założeniu, że plik `powgrader.cpp` jest w tym samym folderze co rozwiązanie:

```
g++ -O3 -static pow.cpp powgrader.cpp -std=c++17
```

Tak skompilowany program wczytuje ze standardowego wejścia dane dotyczące zapytań i wypisuje na standardowe wyjście, w kolejnych wierszach, wyniki kolejnych zapytań `ile_powtorzen`. Format wejścia jest następujący:

- wiersz 1: liczba zapytań z, wliczając wywołanie funkcji `inicjuj`;
- wiersz 2: litera `i`, liczba n, a następnie n liczb oznaczających kolory kolejnych lampek;
- każdy z kolejnych $z-1$ wierszy: litera `p`, a następnie liczby a i b opisujące zapytanie `ile_powtorzen(a, b)`.