Tomasz Waleń
Treść zadania, Opracowanie

Krzysztof Kiljan
Program

Dostępna pamięć: 128 MB.

OI, etap III, dzień pierwszy, 13.03.2013

Łańcuch kolorowy

Mały Bajtuś bardzo lubi bawić się kolorowymi łańcuchami. Zebrał już ich sporą kolekcję, jednak niektóre z nich lubi bardziej niż inne. Każdy z łańcuchów składa się z pewnej liczby kolorowych ogniw. Bajtazar zauważył, że Bajtuś ma bardzo precyzyjne preferencje estetyczne. Otóż Bajtuś uważa jakiś spójny fragment łańcucha za ładny, jeśli zawiera on dokładnie l_1 *ogniw koloru* c_1, l_2 *koloru* c_2, $\ldots$, l_m *koloru* c_m*, a ponadto nie zawiera żadnych ogniw innych kolorów. Atrakcyjność łańcucha odpowiada liczbie (spójnych) fragmentów, które są ładne. Bajtazar metodą prób i błędów odkrył wartości* $c_1, \ldots, c_m$ *i* $l_1, \ldots, l_m$*. Teraz chciałby kupić nowy łańcuch i prosi Cię o napisanie programu, który pomoże mu w zakupach.*

Wejście

Pierwszy wiersz standardowego wejścia zawiera dwie liczby całkowite n *i* m ($1 \leqslant m \leqslant n \leqslant 1\,000\,000$) *oddzielone pojedynczym odstępem. Oznaczają one odpowiednio długość łańcucha i długość opisu ładnego fragmentu. Drugi wiersz zawiera* m *liczb całkowitych* $l_1, \ldots, l_m$ ($1 \leqslant l_i \leqslant n$) *pooddzielanych pojedynczymi odstępami. Trzeci wiersz zawiera* m *liczb całkowitych* $c_1, \ldots, c_m$ ($1 \leqslant c_i \leqslant n$, $c_i \neq c_j$ *dla* $i \neq j$) *pooddzielanych pojedynczymi odstępami. Ciągi* $l_1, \ldots, l_m$ *i* $c_1, \ldots, c_m$ *opisują definicję ładnego fragmentu – musi on zawierać dokładnie* l_i *ogniw koloru* c_i*. Czwarty wiersz zawiera* n *liczb całkowitych* $a_1, \ldots, a_n$ ($1 \leqslant a_i \leqslant n$) *pooddzielanych pojedynczymi odstępami, oznaczających kolory kolejnych ogniw łańcucha.*

W testach wartych 50% punktów zachodzi dodatkowy warunek $1 \leqslant m \leqslant n \leqslant 5000$.

Wyjście

W pierwszym i jedynym wierszu standardowego wyjścia Twój program powinien wypisać jedną liczbę całkowitą – liczbę ładnych spójnych fragmentów w łańcuchu.

Przykład

Dla danych wejściowych:

```
7 3
2 1 1
1 2 3
4 2 1 3 1 2 5
```

poprawnym wynikiem jest:

```
2
```

Wyjaśnienie do przykładu: *Dwa ładne fragmenty tego łańcucha to 2, 1, 3, 1 oraz 1, 3, 1, 2.*

Testy „ocen”:

`1ocen`: *$n = 9$, $m = 3$, dwa ładne fragmenty następują po sobie, ale nie nakładają się;*

`2ocen`: *$n = 10$, $m = 5$, długość ładnego fragmentu przekracza długość całego łańcucha (wynik to 0);*

`3ocen`: *$n = 19$, $m = 7$, trzy ładne fragmenty nachodzące na siebie;*

`4ocen`: *$n = 1000$, $m = 500$, ładny fragment zawiera po jednym kolorze z $\{1, \ldots, 500\}$, a łańcuch to ciąg ogniw kolorów 1, 2, ..., 499, 500, 500, 499, ..., 2, 1 (wynik to 2);*

`5ocen`: *$n = 1\,000\,000$, $m = 2$, ładny fragment zawiera jeden kolor 1 oraz dwa kolory 2, łańcuch to 1, 2, 2, 1, 2, 2, ... (wynik to 999 998).*

Rozwiązanie

Prawidłowe rozwiązanie zadania *Łańcuch kolorowy* nie sprawiło zbyt wielu kłopotów uczestnikom Olimpiady. Aby osiągnąć efektywne rozwiązanie zadania, wystarczy zauważyć kilka prostych faktów.

Szukane ładne fragmenty łańcucha mają zawsze tę samą długość równą $d := \sum_{i=1}^{m} l_i$. Stąd nie musimy rozważać wszystkich możliwych spójnych fragmentów łańcucha (których jest $\Theta(n^2)$), a jedynie $\Theta(n)$ fragmentów o długości dokładnie d. Jeśli zauważymy, że wartość d przekracza długość łańcucha n, to możemy zakończyć obliczenia i zwrócić wartość 0, ponieważ łańcuch jest zbyt krótki, aby zawierał ładny fragment. W tym kroku ukryta była pewna pułapka techniczna, gdyż zgodnie z opisem danych wejściowych wartość d mogła być bardzo duża i konieczne było użycie dla niej typu danych o odpowiednim rozmiarze (na przykład 64-bitowej liczby całkowitej).

Kolejną prostą obserwacją jest fakt, że dwa kolejne fragmenty długości d rozpoczynające się odpowiednio na pozycjach a_i oraz a_{i+1} nie różnią się zbyt wiele, jeśli chodzi o krotności występujących w nich kolorów. Oba fragmenty różnią się jedynie elementami a_i oraz a_{i+d}. Możemy wykorzystać tę obserwację, aby znacznie przyspieszyć testowanie, czy aktualnie badany fragment jest ładny.

Aby dokładniej opisać rozwiązanie, zdefiniujmy funkcję $ile_w(c)$ zliczającą liczbę wystąpień ogniw koloru c w spójnym fragmencie łańcucha $w = a_i, \ldots, a_j$:

$$ile_w(c) = |\{a_p = c : i \leqslant p \leqslant j\}|.$$

Korzystając z poprzedniej obserwacji, zauważamy, że dla dwóch kolejnych fragmentów $w = a_i, \ldots, a_{i+d-1}$ oraz $u = a_{i+1}, \ldots, a_{i+d}$ funkcje $ile_w(c)$ oraz $ile_u(c)$ albo są identyczne (w przypadku gdy $a_i = a_{i+d}$), albo (w przypadku gdy $a_i \neq a_{i+d}$) różnią się na dokładnie dwóch pozycjach: $ile_u(a_i) = ile_w(a_i) - 1$ oraz $ile_u(a_{i+d}) = ile_w(a_{i+d}) + 1$.

Na podstawie definicji ładnego fragmentu definiujemy analogiczną funkcję $oczekiwane(c)$, która dla każdego koloru c zwraca nam liczbę oczekiwanych ogniw l w ładnym fragmencie: $oczekiwane(c_i) = l_i$ dla $1 \leqslant i \leqslant m$ oraz $oczekiwane(c) = 0$ dla pozostałych kolorów.

Oczywiście, fragment $w = a_i, \ldots, a_{i+d-1}$ jest ładny wtedy i tylko wtedy, gdy $ile_w(c) = oczekiwane(c)$ dla $1 \leqslant c \leqslant n$.

Aby przyspieszyć testowanie powyższego warunku, możemy utrzymywać wartość *zleKolory*, równą liczbie kolorów c dla których $ile_w(c) \neq oczekiwane(c)$. Dzięki tej wartości możemy bardzo łatwo testować, czy fragment jest ładny: wystarczy sprawdzić, czy $zleKolory = 0$.

Korzystając z powyższych obserwacji, otrzymujemy następujące rozwiązanie o złożoności czasowej i pamięciowej $O(n)$:

```
function policzLadneFragmenty(A[1..n], L[1..m], C[1..m])
begin
  d := Σ_{i=1}^{m} L[i];
  if d > n then return 0;
  wyznacz tablicę oczekiwane[1..n] na podstawie L[1..m], C[1..m];
  { wyznaczanie wartości ile i zleKolory dla a_1, ..., a_d }
  wyznacz tablicę Ile[1..n], gdzie Ile[c] = |{A[i] = c : 1 ⩽ i ⩽ d}|;
  zleKolory := |{Ile[c] ≠ oczekiwane[c] : 1 ⩽ c ⩽ n}|;
  ladneFragmenty := 0;
  if zleKolory = 0 then ladneFragmenty := ladneFragmenty + 1;
  for i := 1 to n − d do begin
    { wyznaczanie wartości ile i zleKolory dla a_{i+1}, ..., a_{i+d} }
    if A[i] ≠ A[i + d] then begin
      aktywneKolory := {A[i], A[i + d]};
      d_1 := |{Ile[c] ≠ oczekiwane[c] : c ∈ aktywneKolory}|;
      Ile[A[i]] := Ile[A[i]] − 1;
      Ile[A[i + d]] := Ile[A[i + d]] + 1;
      d_2 := |{Ile[c] ≠ oczekiwane[c] : c ∈ aktywneKolory}|;
      zleKolory := zleKolory − d_1 + d_2;
    end
    if zleKolory = 0 then ladneFragmenty := ladneFragmenty + 1;
  end
  return ladneFragmenty;
end
```

Testy

Do oceny zadania przygotowano 10 zestawów testowych, z których każdy zawierał od 4 do 5 testów. Testy zostały podzielone na trzy rodzaje:

- ciągi zupełnie losowe,
- ciągi, których ładne podciągi nachodzą na siebie,
- ciągi zawierające rozłączne ładne podciągi.

Ponadto w każdej grupie był test zawierający ładny podciąg zaczynający się na pierwszej pozycji ciągu.