Omogen (omogen)

Author: Alexandru Gheorghies

Developer: Stefan Dascalescu

Solution

Pentru obinerea punctajelor din primul subtask, putem folosi diverse metode brute-force care calculeaz cel mai mare divizor comun pentru fiecare pereche de valori.

O proprietate foarte important a celui mai mare divizor comun este aceea c dac $cmmdc(a, b) = 1$, atunci cele dou numere nu au niciun factor prim comun. Dac extindem aceast proprietate pentru o întreag subsecven de valori, atunci putem trage concluzia c oricare ar fi un numr prim x, nu poate s apar în reprezentarea în factori primi a mai mult de un numr din acea subsecven, deoarece în caz contrar, ar exista o pereche de tip (a_2, b_2) cu cmmdc-ul diferit de 1.

Astfel, vom precalcula toate numerele prime mai mici decât 10^7, precum i descompunerea în factori primi, pe care o vom stoca într-un mod compresat pentru a evita folosirea unei cantiti prea mari de memorie. Acest lucru se poate face folosind un algoritm similar cu ciurul lui Eratostene. Pentru diverse punctaje pariale se pot folosi algoritmi care afl divizorii în mod obinuit.

În final, putem folosi un algoritm de tip two-pointers folosind un vector de frecven care pstreaz informaii despre fiecare divizor prim care apare, iar atunci când avem un numr prim care apare pentru a doua oar în ir, vom scoate valori din captul din stânga pân când aceast proprietate nu mai este adevrat.

In order to obtain the scores from the first subtask, we can rely on various brute force methods which compute the greatest common divisor for every pair of numbers in the subarray.

A very important property of the greatest common divisor is that if $gcd(a, b) = 1$, then a and b don't have any common prime factors. If we expand this property for an entire subarray, we can conclude that for every prime x which is present in the prime factorization of at least one of the numbers, then it can't show up in more than one number's factorization, because if this property wouldn't be true, there would be a pair (a_2, b_2) such that its GCD would be greater than 1.

Thus, we will precompute all prime numbers less than 10^7, together with the prime factorization which will be stored in a compressed manner in order to avoid using way too much memory. This can be done using an algorithm similar to the sieve of Eratosthenes.

Last but not least, in order to compute the number of homogenous subarrays, we will rely on the information found previously in order to track whether each prime number shows up or not using two pointers and a frequency array, and as we find a prime number which shows up twice, we will remove integers from the left.

Majorat (majorat)

Author: Vlad-Mihai Bogdan

Developer: Vlad-Mihai Bogdan

Solution

In this editorial, we will be using the term subsequence as contiguous subsequence (also known as a substring).

If we consider an array of length N only containing ones, any contiguous subsequence of the array will have a majority element, for a total of $\frac{N\cdot(N+1)}{2}$ subsequences. This gives us some kind of lower bound for the length of the array. We will try to find a solution with length of order $\sqrt{N}$.

We will use the following building block $x_0x_1x_0x_1\ldots x_0x_1$. In this array, any subsequence of odd length will have a majority element. More then that, any subsequence of even length will not have a majority element. If the length of this array is N, the number of subsequences having a majority element is $\lfloor\frac{(N+1)\cdot(N+1)}{4}\rfloor$.

For two building blocks $x_0x_1\cdot x_0x_1\ y_0y_1\cdot y_0y_1$, if x_0, x_1, y_0, y_1 are all distinct, only subsequences inside the same building block will create subsequences having a majority element.

With this building block in mind, we only need to write K as a sum of some values $l_0+l_1+l_2+\cdot+l_{m-1}$, where $l_i=\lfloor\frac{t\cdot t}{4}\rfloor$. An easy way to do this is to always try to add the largest value that does not exceed K.

It is easy to show that this solution will not make more than $3\cdot\sqrt{K}$ for large enough values of K. This is enough to get full marks, since the maximum accepted length of the array in this problem is around $4\cdot\sqrt{K}$.

Many more other solutions can be found to this problem, some with even lower lengths.

S considerm un bloc de lungime N având urmtorul format $x_0x_1x_0x_1\cdot x_0x_1$. În cadrul acestui bloc, orice subsecven de lungime impar va avea element majoritar, iar orice subsecven de lungime par nu va avea element majoritar. Numr de subsecvene de lungime impar este $\lfloor\frac{(N+1)\cdot(N+1)}{4}\rfloor$.

Mai mult, dou bloc-uri de acest tip cu valori distincte dou câte dou nu vor interaciona (nu se vor forma subsecvene care încep într-un bloc i se termin în altul). Prin urmare, vrem s îl scrie pe K ca sum de numere de forma $\lfloor\frac{(N+1)\cdot(N+1)}{4}\rfloor$.

O strategie care ofer lungimi suficient de mici pentru irul nostru este s alegem mereu cea mai mare lungime a unui bloc care poate fi adugat la ir.

Cu aceast soluie, este uor s artm c lungime este maxim $3\cdot\sqrt{K}$ pentru valori ale lui K suficient de mari, care se încardreaz lejer în limita de $4\cdot\sqrt{K}$ impus de problem.