475D – CGCDSSQ

https://codeforces.com/contest/475/problem/D

Обяснение на алгоритъма за решаване на задачата

Изглежда, че какъвто и подход за решение да опитаме, всеки един от тях ще се основава на едно и също наблюдение. Нека въведем две числови редици: $a_0, a_1, \ldots, a_{n-1}$ от произволни цели положителни числа и $x_1, x_2, \ldots, x_{n-1}$, където $x_0 = a_0$, $x_i = gcd(x_i, a_i)$. Наблюдението е, че броят на различните стойности в редицата $\{x_i\}_0^{n-1}$ не надвишава $1 + log_2 a_0$. Това се доказва лесно като се вземе предвид, че редицата $\{x_i\}_0^{n-1}$ не е растяща и стойността на най-големия общ делител, в случай че намалява, може да стане най-много половината от предишната стойност. След като имеме това наблюдени предвид искаме да прекалкулираме броя на стойностите на GCD през всички интервали и да ги запомним в някакъв удобен за съхранение контейнер. Може да търсим местата, където функцията $GCD(a_l, \ldots a_r)$ ще намалее и да актуализираме лявата част на интервала a_l. Между точките, в които намалява, тя ще остане константна, така че може да добавим размера на този *плосък* интервал към резултата от тази конкретна стойност за GCD. За намирането на местата, където функцията променя стойноста си, може да се подходи по различни начини. Един от тях, който ще разгледаме използва структурата от данни-разредена таблица (Sparse Table). Sparse Table ще ни даде информация за стойността на GCD за всеки посочен интервал за константно време. Чрез тази информация ще може да осъществим бинарно търсене, за да намерим местата, на които функцията GCD намалява. Времевата сложност за изчисление на точка в която функцията намалява е $log(n)$, и ще има максимум $log(n)$ такива търсения. Това ще се извърши общо n на брой пъти. Общата времева сложност ще достигне $O(n \times log^2(n))$, която ще надвиши с малко сложността за построяване на разредената таблица. Добавяме и броя на заявките и получаваме финалната времева сложност на описания алгоритъм $O(n \times log^2(n) + m)$.