АНАЛИЗ НА РЕШЕНИЕТО НА ЗАДАЧА
МОДНИЦА

Трябва да се изчисли сумата от най-големите общи делители (НОД) на всички подинтервали на даден масив. Отговорът се извежда по модул $10^9 + 7$.

Ключово наблюдение.

За фиксиран ляв край на подинтервала *i* да разгледаме последователността от стойности $gcd(a_i, a_{i+1}, \ldots, a_j)$ при увеличаване на десния край *j*. Тази последователност:

- не нараства;
- приема не повече от $O(logA)$ различни стойности, където $A = max\ a_i$.

Това е свързано с факта, че всеки път, когато НОД намалява, той се дели поне на едно просто число и за $O(logA)$ стъпки достига 1.

Идея на решението.

Ще обработваме масива отляво надясно. За текущата позиция ще съхраняваме компресиран списък (или речник) от двойки (g, cnt), където:

- g — стойността на НОД за подинтервалите, завършващи на текущата позиция;
- cnt — броят на тези подинтервали.

При преминаване към нов елемент *x*:

- създаваме ново състояние, започващо с подинтервала [*x*] (т.е. добавяме двойката $(x, 1)$);
- за всяка предишна стойност на g изчисляваме $gcd(g, x)$ и добавяме съответния брой подинтервали към новото състояние;
- еднаквите стойности на НОД автоматично се обединяват (в речника — чрез сумиране на броячите).

Такъв списък винаги съдържа не повече от $O(logA)$ елемента.

Изчисляване на отговора.

На всяка стъпка за всяка стойност на g в текущото състояние добавяме към отговора величината $g \cdot cnt$, взета по модул $10^9 + 7$.

Сложност.

Алгоритъмът извършва $O(n\ logA)$ операции за изчисляване на НОД, което се вписва в ограниченията за $n \leq 200\ 000$.

Подходи за различните подзадачи.

- **Подзадача 1** ($n \leq 250$). Използва се директно изброяване на всички подинтервали с троен вложен цикъл: за всяка двойка с граници (l, r) се изчислява НОД на всички елементи в интервала. Такова решение има сложност $O(n^3\ logA)$ и е достатъчно за най-малките тестове. Реализация в *trendy_22.cpp*.

- **Подзадача 2** ($n \leq 1000$). Подобряваме предишния метод: за фиксиран ляв край *l* последователно разширяваме десния край *r* и обновяваме текущия НОД по формулата НОД = *gcd*(текущ НОД, a_r). Това дава сложност $O(n^2\ logA)$, което се вписва в ограниченията за $n \leq 1000$. Реализация в *trendy_37.cpp*.

- **Подзадачи 3 и 4** ($n \leq 200\ 000$). Използва се ключовото наблюдение: при фиксиран десен край броят на различните стойности на НОД сред всички подинтервали, завършващи в тази позиция, не надвишава *O(logA).* Поддържаме списък (или речник) от двойки (*g, cnt*), където *g* — стойност на НОД, а *cnt* — брой подинтервали с такъв НОД. При преминаване към следващия елемент обновяваме всички стойности на НОД и обединяваме еднаквите. Сложност — *O(n logA).* Реализация в *author.cpp.*

Обобщение. Основната идея на пълното решение е ефективното компресиране на множеството от възможни НОД-ове за подинтервалите, завършващи в текущата позиция. Благодарение на логаритмичния брой различни стойности на НОД алгоритъмът работи бързо дори при големи *n*.

Автор: Кинка Кирилова-Лупанова