К-то – решение

Атанас Димитров

Подзадача 1

Ще наричаме елемент **важен** в сегашния момент, ако той е един от елементите на позиции $1, 1+k, 1+2k, ...$, т.е. задачата изисква намирането на сумата на **важните** елементи.

Първата подзадача изисква директна реализация на условието на задачата. Ще поддържаме масив, в които ще добавяме a_i на всяка стъпка и ще сортираме наново. След всяко добавяне и сортиране, можем да обходим **важните** елементи и да намерим тяхната сума.

Общата сложност е $O(n^2 \log(n))$.

Подзадача 2

Можем да съобразим факта, че ако $k = n$, то n-красотата на мултимножеството е равна на най-малкият му елемент. Тогава, за решаване на задачата, можем да държим мултимножеството в приоритетна опашка, която да "питаме" за най-малкия вече добавен елемент.

Общата сложност е $O(n \log(n))$.

Подзадача 3

Ще разгледаме по-генерално решение, което решава задачата ефективно за $k \leq \sqrt{n}$.

Нека започнем като сортираме всички числа, които ще присъстват в мултимножеството в края. Ще кръстим тази сортировка b и ще я разделим на $O(\sqrt{n})$ с приблизително равни размери, като i-тия "бъкет" отговаря на интервала $[l_i, r_i]$.

"Бъкетите" ще използваме, за да поддържаме k-красотата на мултимножеството бързо. Можем да забележим, че за да определим кои са **важните** елементи в даден момент във всеки "бъкет" ни е единствено нужно да знаем броя на добавените елементи в мултимножеството в "бъкетите" вляво и в частност остатъка на този брой при деление на k. Допълнително ако даден "бъкет" не се променя, то сумата на **важните** елементи зависи единствено от този предишен остатък.

Така можем да достигнем до следното решение:

Когато добавяме нов елемент, който принадлежи в "бъкет" i, смятаме за всеки възможен предишен остатък, колко би била сумата и колко биха били **важните** елементи, ако започвахме от този остатък. Тъй като всеки елемент в даден "бъкет" може да е **важен** само по един начин(за фиксираното k), то тази стъпка може да бъде реализирана във време $O(k) = O(\sqrt{n})$.

Когато искаме да разберем сегашната k-красота, започваме да разглеждаме "бъкетите" последователно започвайки от 0. Ще започнем от предишен остатък 0(защото няма предишни "бъкети") и ще го променяме със съответния брой **важни** елементи, които срещаме в този "бъкет". Имайки предишния остатък за всеки "бъкет", можем лесно да намерим k-красотата като сумираме преизчислените стойности. Тази стъпка може да бъде реализирана в $O(\sqrt{n})$.

Общата сложност е $O(n\sqrt{n})$.

Довършване на цялата задача

За довършване на задачата трябва да се справим единствено със случаят $k > \sqrt{n}$.

Нека забележим проблема с миналото решение – ако $k > \sqrt{n}$, то остатъците които трябва да разглеждаме стават много повече и наивното пресмятане става неефективно. Въпреки това, можем с лека модификация да се справим с този проблем – тъй като k е голямо, най-големия брой **важни** елементи е $O(\frac{n}{k}) = O(\sqrt{n})$. Допълнително е важно да се отбележи, че ако $k > \sqrt{n}$, то във всеки "бъкет" ще има най-много по един **важен** елемент.

Тогава ако за всеки "бъкет" пазим в масив с динамичен размер(например std::vector) всички добавени елементи, можем да обиколим "бъкетите" поддържайки предишния остатък както в миналата подзадача и ако се налага да прибавяме единствения **важен** елемент в този "бъкет", като пресметнем индекса му.

Общата сложност е $O(n(\frac{n}{k})) = O(n\sqrt{n})$, от което следва че цялата задача е решима в $O(n\sqrt{n})$.