Алгоритмы и Структуры Данных ДЗ-4

Гарипов Роман М3138

19.10.2019

Задача №1

Воспользуемся методом двоичных прыжков. Заведём переменную $id = 0$ и $\Delta = 1$. Проверим, если $a_{id} < x$, то совершим прыжок на Δ, $id+ = \Delta$, после чего увеличим значение этой переменный на 2. Таким образом всего мы пропрыгаем $2^0 + 2^1 + \cdots + 2^{k-1} + 2^k = 2^{k+1} - 1 \geq p$. Видно что всего будет $\mathcal{O}(\log(p))$ прыжков, ведь на предыдущем щаге сумма была $2^k - 1 < p$. Поэтому $k = \lceil \log(p) \rceil - 1 = \mathcal{O}(\log(p))$. Далее найдем x на отрезке $[id - \frac{\Delta}{2}; id]$ бинарным поиском. Он отработает за $\log(id - id + \frac{\Delta}{2} + 1) = \log(\frac{\Delta}{2} + 1) \leq k = \mathcal{O}(\log(p))$. Обе фазы алгоритма работают за $\mathcal{O}(\log(p))$, следовательно сам алгоритм имеет ассимптотику $\mathcal{O}(\log(p))$.

Задача №2

Будем решать методом двух указателей. Только для правого конца отрезка будем хранить не 1 указатель, а 2.
Пусть у нас есть наш актуальный отрезок(назовём это *Начальная Фаза*) $s_1 = [l; r_1]$ и $s_2 = [l; r_2]$. На s_1 и s_2 должно выполнятся такое условие, что на них и на всех отрезках которые начинаются в l и кончаются где-то между r_1 и r_2, должно быть ровно k различных элементов, а на отрезке $[l; r_2 + 1]$ уже $k + 1$ различный элемент.
Тогда, попробуем подвинуть l, если количество различных элементов на $[l; r_1]$ не уменьшилось(тогда и на $[l; r_2]$ не уменьшилось) добавим к ответу $r_2 - r_1 + 1$, ведь это все отрезки которые начинаются в l и имееют ровно k различных элементов внутри. Ну и двигаем так увеличивая ответ, пока кол-во различных не уменьшится.
Когда это кол-во уменьшилось, свойства наших отрезков перестали выполнятся, надо сформировать новые для которых это свойство выполняется. Поставим $r_1 = r_2 + 1$, а r_2 будем сдвигать вправо до тех пор, пока количество различных элементов не изменяется, то есть встречаем элементы которые уже были. Назовём все это *Переходом*.
Для этого будем поддерживать сколько раз встречается каждый элемент на актуальном отрезке(массив подсчёта). Соотвественно, нужно поддерживать эту информацию актуальной, поэтому когда двигаем соответствующие границы отрезка - будем увеличивать или уменьшать кол-во элементов на отрезке для только что добавленного элемента, или для только что удаленного.
Осталось только уточнить как проверять сколько различных чисел на отрезке, ну для этого будем хранить(и соответственно изменять при добавлении/удалении элемента) количество не нулей в нашем массиве подсчета. Это и будет количеством различных чисел на отрезке.
После того как мы поставили r_1 и r_2 на новое места, мы пришли к той же *Начальной Фазе*, продолжим повторять наш *Переход*, пока не пройдем весь массив указателями.
Получаем решение трёмя указателями, каждый из которых проходит по

каждому элементу не более 1 раза, следовательно ассимптотика $\mathcal{O}(n)$.

Задача №3

Для определённости возьмём $n < m$. Будем делать бинарный поиск по длине префикса в меньшем массиве a. Пусть сейчас наш отрезок $[l; r]$. Нужно понять какую границу сдвинуть в $mid = \frac{l+r}{2}$. Если $a[mid] > b[k - mid]$ то сделаем $r = mid$, иначе $l = mid$. Таким образом, наименьшее из этих двух чисел всегда будет оставаться на новом отрезке в соответствующем массиве. Так мы будем постоянно менять колво элементов на префиксе в массиве a, пока не уткнемся в границу. Ответом будем $max(a[r], b[k - r])$