Домашняя работа №5

Шибаев Александр Б05-222

Декабрь 2022

1 Первая задача

Давайте посортим все футблоки так, чтобы если две футболки имеют одинаковую цену, то они упорядочены по качеству. Это мы делаем за $O(n \log n)$. Теперь создадим декартово дерево, в котором ключем будет количество денег, и будет храниться индекс человека, соответствующего данной вершине и число купленных футболок. Ассиптотика $O(k \log k)$. Теперь пройдемся по отсортированному массиву футболок слева направо. Тогда $i-$ футболку могут купить покупатель у которых хотя бы c_i денег. Теперь сплитанем дерево по c_i, получим людей, у которых денег $< c_i$ и людей, у которых денег $\geq c_i$. И теперь для поддерева R, с людьми, у которых $\geq c_i$ денег увеличим количество купленных футблок на 1, у количество денег уменьшим на c_i отложенными операциями. Теперь давайте переложим все вершины из R с ключами, меньшими, чем наибольший ключ(начинаем перекладывать сначала с самым маленьким ключем) из L(людей с $< c_i$ денег) в L и смержим два получившихся дерева. $Merge$ работает за $O(\log k)$. Теперь посчитаем, сколько максимум перекладываний мы могли сделать. Пусть наибольший ключ в L был M, тогда в R был ключ m, и выполняются два неравенства: $M < c_i \leq m$ и $m - c_i \leq M \Rightarrow$M < m < 2 c_i. Т.к. $m \geq c_i(m \in R) \Rightarrow$m $\in [c_i, 2c_i]$, а занчит после покупки футблоки $m \in [0, c_i]$. Поэтому каждую вершины мы можем перекладывать максимум $\log C$ раз.
Т.к. футболок всего n, то нам нужно будет проделывать описанные операции n раз: $Merge+Split$ за $O(n \log k)$ и все перекладывания за $O(k \log k \log C)$(т.к. перекладывание за $O(\log k)$). И в конце идем по ДД и записываем ответ для каждого покупателя - $O(k \log k)$. Суммарная ассимптотика $O((n \log n) + (n \log k) + (k \log k \log C))$.

2 Вторая задача

Допустим существует, тогда n элементов мы можем положить в это дерево за $O(n)$. Давайте выведем это дерево - сначала запустим рекурсивный вывод от левого сына, потом выведем саму вершину, потом запустим рекурсивный вывод от правого сына. Таким образом получим отсортированный массив. Но мы сначала потратили $O(n)$ чтобы построить дерево, а потом $O(n)$, чтобы его вывести $\Rightarrow$ O(n)$-$, , O(n $\log n$) - противоречие.

3 Третья задача

Давайте хранить в AVL удаленные вершины с их изначальными номерами, а так же для каждой вершины хранить количество вершин в ее правом поддереве. Тогда если мы хотим удалить вершину с номером k в новой нумерации, то мы находим ее номер в старой нумерации и кладем в AVL. Как это делать - пусть мы находимся в вершине v нашего дерева удаленных вершин. У v в левом поддереве l вершин, тогда если $k + l + 1 \geq$ номер вершины v(в старой нумерации), то это значит, что в старой нумерации вершина, у которой сейчас номер k имела номер больший, чем номер вершины v, т.к. мы удалили все вершины, которые лежат в левом поддереве v и саму v, и значит номера вершин, у которых номер был больше, чем номер v уменьшился на размер левого поддерева $v + 1$. Поэтому мы увеличиваем k на $l + 1$ и идем в правое поддерево v, а иначе (если $k + l \leq$ номер вершины v) просто идем в левое поддерево v, а если $k + l ==$ номер вершины v, то мы нашли саму вершину и возращаем ответ. Такми образом мы получим номер в исходной нумерации и в зависимости от типа запроса будем делать действие - если запрос на удаление, то добавляем вершину с пересчитанным номером в дерево, иначе выводим вершину с пересчитанным номером.