Задача A. Арбуз с хлебом

Для того, чтобы масса арбуза делилась на количество друзей необходимо съесть $W\%N + N \cdot k$ килограмм арбуза, минимальным, в данном случае, значением является значение $W\%N$. В случае, если Аяз должен съесть больше чем может, то выводиться -1. Во всех указанных формулах рассматривается деление на цело и знак % в качестве операции взятия остатка от деления.

Задача B. Цветные карточки

Из жадных соображение очевидно, что в итоговых массивах значения должны быть отсортированы. Так же очевидно, что если значения должны быть отсортированы, то оптимальным разбиением будет взять массив исходный массив, отсортировать его и положить все элементы на четных позициях в один массив, а на нечетных в другой. Допустим это не так, к примеру, есть всего 4 элемента a0 < a1 < a2 < a3 И в первый массив входят элементы a0, a1, а во второй a2, a3, тогда из неравенства выше очевидно, что a2 - a0 + a3 - a1 = a2 + a3 - a0 - a1 > a1 - a0 + a3 - a2 = a1 + a3 - a0 - a2

Задача C. Скобочная последовательность

Формализуем задачу, нас просят сказать, правда ли, что существует циклический сдвиг строки, являющийся ПСП (правильной скобочной последовательностью). Давайте возьмем строку s из входных данных, и запишем её дважды друг за другом s + s, таким образом нас просят сказать, правда ли, что в новой строке, существует подстрока, являющаяся ПСП. Переберем все подстроки, подстрока (l, r) является хорошей, если количество открывающихся и закрывающихся скобок на отрезке одинаково, а так же на подстроке не существует префикса, на котором количество закрывающихся скобок больше количество открывающихся. Проверить условие с количеством скобок на префиксе можно насчитав баланс ПСП, давайте заменим все открывающиеся скобки на +1, а закрывающиеся на -1, таким образом проверка про существование плохого префикса, сводится к поиску минимума в массиве, а это можно сделать структурой set за O(log(n))

Задача D. И снова седой мекс

Первоначально, давайте найдем отрезок (1, r) такой, что mex на этом отрезке больше W. Это можно сделать за O(n) с массивом, будем итерироваться по префиксу и когда будем встречать число V будем делать +1 в массив used на позицию V, так же будем вести второй указатель по массиву used, изначально указатель будет указывать на позицию cur = 0, будем сдвигать его каждый раз, когда used[cur] != 0. Этот указатель будет являться мексом для отрезка (1, r), суммарно мы сдвинем его не больше чем n раз, таким образом получаем ассимптотику O(n) для нахождения отрезка. Далее давайте научимся переходить от отрезка (1, r) к отрезку (2, r'). Если значение на позиции 1 не встречается нигде более на отрезке (2, r) и значение на этой позиции меньше W, то наш отрезок изменится следующим образом, найдем ближайшего справа от индекса r соседа со значением равным значению на позиции 1 и скажем, что r' равно этой найденной позиции. Если же значение на позиции 1 меньше чем W или оно встречается несколько раз на отрезке (1, r) то, значение r' не изменится и будет так же равно r. Поиск соседа справа от индекса и проверка сколько раз элемент встречается на отрезке может быть организованна при помощи бинарного поиска, заведем массив position, где position[i] будет хранить все позиции вхождения элемента i в отсортированном порядке, бинарный поиск по этому массиву решает все поставленные задачи.

Альтернативный разбор https://www.overleaf.com/read/zpcnfvyybvhn

Задача E. Построй свой массив!

Считаем все запросы на добавление и создадим конечный массив сразу же, а после будем отвечать на запросы. Построим дерево отрезков, с отсортированными векторами в вершине(merge tree), где в вершине, отвечающей за отрезок (l, r) лежат отсортированные элементы с этих позиций. В таком случае ответ на запрос количество элементов меньше либо равных сводится к бинарному поиску в log вершинах дерева отрезков, т.е к ассимптотике $O(q * log(q)^2)$ https://cf.mfyan.com/blog/entry/75304 статья по структуре данных