# Почти наверное

### Подгруппа 1.

Для каждого префикса давайте переберем пару чисел, которые будут различаться. После этого выкинем их из массивов и посчитаем сколько операций потребуется, чтобы сделать массивы равными. Число операций равно сумме элементов первого массива минус сумма элементов второго массива. А уровнять их можно только если для каждого $i$ выполнено $a_i \geqslant b_i$. Такое решение работает за $O(n^4)$.

### Подгруппа 2.

Давайте улучшим решение прошлой подгруппы. Мы хотим выкинуть по элементу из обоих массивов так, чтобы разница между ними была как можно больше. Давайте отсортируем оба префикса. Заметим, что если мы можем удалить $a_i$ и $b_j$, то мы можем удалить $a_{i-1}$ и $b_j$, а также $a_i$ и $b_{j+1}$. Тогда нам необязательно перебирать $O(n^2)$ пар для удаления, а можно перебирать элемент первого массива, а элемент второго находить двигая указатель. Такое решение работает за $O(n^3)$.

### Подгруппа 3.

Снова улучшим решение прошлой подгруппы. Нужно быстрее проверять условие $a_i \geqslant b_i$. Для этого нужно, заметить что $b_i$ элемент сравнивается либо с $a_i$, либо с $a_{i+1}$. При чем оба префикса будут разбиваться на $O(1)$ отрезков, в которых нужно делать сравнения одного из двух типов. Давайте префиксными суммами предподсчитаем выполняются ли сравнения каждого из типов, после чего можно будет проводить проверку корректности за $O(1)$. Такое решение работает за $O(n^2 \log n)$.

### Идеи для полного решения.

Давайте будем смотреть на два массива, как на набор отрезков $[b_i, a_i]$. Заметим, что нам никогда не выгодно делать в итоговом ответе $a_i < b_i$. Тогда посмотрим как будет выглядить итоговый ответ. Удалим все индексы для которых $a_i = b_i$. Отсортируем оставшиеся числа по возрастанию $a_i$, тогда $a_1 \leqslant a_2 \leqslant \ldots \leqslant a_n$ и $b_1 < a_1, b_2 < a_2, \ldots b_n < a_n$. Нетрудно понять, что нужно выкинуть $a_n$ и $b_1$.

Каким условиям должны удовлетворять отрезки, чтобы после удаления $a_n$ и $b_1$ все было валидно? $b_2 \leqslant a_1, b_3 \leqslant a_2, \ldots, b_n \leqslant a_{n-1}$. Если говорить об этом как о отрезках это значит, что все они образуют связную компоненту. Тогда ответом является сумма длин отрезков минус длина в координатах самой длинной компоненты.

### Подгруппа 4-5.

Используя это можно понять, что ответ в 4 подгруппе равен сумме длин отрезков минус длина максимального отрезка. Для пятой группы нужно поддерживать текущую компоненту, при возможности ее расширять и запоминать самую длинную компоненту до этого.

### Полное решение.

В реализации полного решения давайте поддреживать текущие компоненты отрезков. При переходе к новому префиксу нужно уметь сливать эти компоненты, если они пересекаются с новым отрезком. Все это можно легко поддерживать в std::set. Время работы решения $O(n \log n)$