Бабушкин А.

1 Обязательные задачи

1.

a) Нашли медиану, сделали массив из элементов `|pos(x) - pos(median)|`, нашли там k-ую порядковую, выписали все элементы, которые меньше этой k-ой.
Update: Найдём элемент, для которого pos(x) = pos(median) - k / 2 и такой, что pos(x) = pos(median) + k / 2. Тогда все элементы между ними и есть k ближайших к медиане.
b) Нашли медиану, сделали массив из элементов `|x - median|`, нашли там k-ую порядковую, выписали все элементы, которые меньше этой k-ой.

2.

a) $f = \sum_i (w_i(x_i - x^*)^2), f' = 0$ при $x^* = \frac{w_i x_i}{n}, O(n)$.
b) По каждой из осей можно решать независимо, то есть у нас есть задача $\sum_i (w_i |x_i - x^*|) \to min$, а это просто найти взвешенную медиану, $O(n)$.
c) Повернём на $\frac{\pi}{4}$, теперь вместро суммы $|x_i - x^*| + |y_i + y^*|$ у нас $max(|x_i - x^*|, |y_i - y^*|) \to$ ответ на задачу – максимум из максимумов, то есть снова независмые задачи по каждой из координат. Надо решить задачу $max_i(w_i|x_i - x^*|) \to min$. Это можно представить как отрезки, расширяющиеся со скоростью w_i, и нам надо найти первый момент времени, когда их пересечение непусто. Давайте посмотрим, где будет находиться самая левая из правых границ отрезков в каждый момент времени. Это возврастающая функция и она кусочно-линейная, её можно построить за $O(sort{+}n)$, отсортировав точки по w_i и построив пересечение полуплоскостей. Аналогично строим для самой правой из левых границ. Нам нужен момент времени, когда самая правая из левых $\geq$ самой левой из правых $\to$ надо найти, где две кусочно-линейные функции пересекаются, это делается двумя указателями. Итого $O(sort + n)$.

3. Отсортируем массив. Теперь если мы поставим две точки куда-то, то одна будет отвечать за некоторый префикс этого массива, а другая – за оставшийся суффикс. Пусть мы решили, за какой префикс должна отвечать первая точка. Тогда нам надо её поставить во взвешенную медиану этого префикса, а вторую – во взвешенную медиану оставшегося суффикса. Если мы будем перебирать этот предполагаемый префикс слева направо, то эти медианы можно поддерживать, как и суммы $\to$ можно решить за $O(sort + n)$ с помощью трёх указателей.
Update: Насчитаем префиксные суммы по w_i в отсортированном массиве. Тогда первый указатель будет показывать в такой первый элемент i, что префсумма на i-ом префиксе $\geq$ половины префсуммы на j-ом префиксе, где j мы как раз перебираем слева направо. Аналогично поддерживаем указатель на суффиксе с помощью суффиксных сумм. Поддерживать легко: пусть мы увеличили j, тогда `while(pr[i] < pr[j] / 2) i++`. Поддерживать сумму взвешенных расстояний тоже легко: пусть слева от указателя i сумма весов w_1, а справа w_2. Тогда если мы сдвинем i в i + 1, то сумма расстояний изменится на `(q[i + 1] - q[i])`$(w_1 - w_2)$. w_1 и w_2 тоже легко поддерживать.

4. **Update:** Отсортируем p_i за O(n + m) подсчётом, затем будем действовать так: найдём элемент, который будет стоять в отсортированном массиве на позиции $p_{m/2}$. Тогда для всех i < m / 2 их элементы будут лежать слева от найденного и наоборот. Рекурсивно спустимся в левую часть и найдём все p_i-ые статистики, где i < m / 2, потом в правую аналогично. На каждом уровне рекурсии мы делаем $\Theta(n)$ действий, а уровней $\Theta(\log m)$. Итого $O(n \log m + n + m) = O(n \log m + m)$ действий. чтд.

5.
   a) $T(n) = \Theta(n) + T(\frac{n}{7}) + T(\frac{3n}{4}), \frac{1}{7} + \frac{3}{4} < 1 \to T(n) = \Theta(n)$
   b) $T(n) = \Theta(n) + T(\frac{n}{3}) + T(\frac{3n}{4}), \frac{1}{3} + \frac{3}{4} > 1 \to T(n) = \Theta(\alpha^n), \alpha > 1$

2 Дополнительные задачи

1. При фиксированном y^* и движении x^* слева направо функция будет выпуклой вниз, и то же самое для y^* $\to$ давайте сделаем два вложенных тернарных поиска.
   Update: Доказывать сложно и грустно, давайте лучше сделаем линейное решение. Мы умеем решать для случая, когда точки невзвешенные, рандомным алгоритмом находя минимальную охватывающую окружность. Давайте сделаем ровно то же самое, изменив две вещи:
   1) Метрикой теперь будет не расстояние между точками, а расстояние$\cdot w_i$, поэтому принадлежность точки окружности мы будем проверять сравнением радиуса с этим новым взвешенным расстоянием.
   2) Когда мы хотим построить окружность, на которой лежит три выбранных точки, мы хотим новую точку O такую, что расстояние от O до трёх данных равно. Раньше ГМТ O таких, что dist(O, A) = dist(O, B) было серединным перпендикуляром к отрезку AB, а теперь $w_A \cdot dist(O, A) = w_B \cdot dist(O, B) \to \frac{dist(O,A)}{dist(O,B)} = const$. Это окружность Аполлония. Чтобы найти подходящую точку O, мы должны будем пересечь две окружности Аполлония.
   Эти модификаций с метрикой достаточно, чтобы в остальном алгоритм не изменился и в итоге мы решили задачу за O(n).

2. Пусть `p[i][j]` – позиция, куда лучше всего поставить последнюю точку, если мы хотим оптимально разбить префикс длины `i` с помощью `j` точек. Заметим, что `dp[i][j]` $\leq$ `dp[i + 1][j]` $\to$ можно сделать Divide-and-conquer, пересчитывая p по слоям (разбить с помощью j точек – это выбрать последнюю и потом разбить оставшийся префикс с помощью j - 1 точки), решение за $O(nk \log n) = O(n \log n)$ в нашем случае.

3. Давайте научимся делать merge двух куч за $O(h)$, где h – высота максимальной из них. Как это сделать? Пусть корень первой кучи < корня второй. Тогда сделаем его корнем новой кучи, вторую кучу подвесим к нему слева, а справа подвесим merge его детей. Тогда $T(h) = 1 + T(h - 1) = \Theta(h)$. Теперь давайте хранить две кучи – основную и дополнительную, размера $\log n$. Add делаем

во вторую за $O(\log \log n)$. Когда она становится больше $\log n$, мёржим её с первой за $\Theta(\log n)$, таких операций будет сделано не больше $\frac{n}{\log n} \to$ суммарно они работают за $\Theta(n)$. Когда мы извлекаем элемент, то просто хотим извлечь минимальный из двух куч. Таким образом, Add работает за $O(\log \log n)$, а асимптотика других функций не испортилась. Успех.