Разбор задачи «Наша Таня громко плачет»

Если $k = 1$, то ответ на задачу, очевидно, $(n-1) \cdot A$, в противном случае будем жадно уменьшать количество файлов жадно.

В каждый момент времени мы можем находиться в одной из трёх ситуаций:

1. Если $n < k$, то нам ничего не остается, кроме как $n - 1$ раз уменьшить число за A, мы можем сделать это за $O(1)$ формулой.

2. Если $n > k$ и n не кратно k, то нам ничего не остается, кроме как $(n \bmod k)$ уменьшить число за A, эту часть тоже можно сделать за $O(1)$ формулой.

3. Если n кратно k, то нам всегда выгодно сделать переход к $\frac{n}{k}$ за $min(B, (n - \frac{n}{k}) \cdot A)$. Если $B < (n - \frac{n}{k}) \cdot A$ корректность очевидна. Иначе предположим мы не сделали переход к $\frac{n}{k}$ сейчас, а сделали его на отрезке $(\frac{n}{k}; n)$ из числа $n - i \cdot k$, тогда мы заплатили $min(B, (n - i \cdot k - \frac{n}{k} + i) \cdot A) + i \cdot k \cdot A$. Это $(n - i \cdot k - \frac{n}{k} + i) \cdot A + i \cdot k \cdot A = (n - \frac{n}{k}) \cdot A - i \cdot (k-1) \cdot A + i \cdot k \cdot A = (n - \frac{n}{k}) \cdot A + i \cdot A$ или $B + i \cdot k \cdot A$, что не выгоднее перехода сразу к $\frac{n}{k}$ и последующему спуску к числу $\frac{n}{k} - i$. Данный переход тоже делается за $O(1)$.

Поскольку в каждой ситуации мы можем побывать не более $\log_k n$ раз асимптотика решения $O(\log_k n)$.

Разбор задачи «Новое — это хорошо забытое старое»

Рассмотрим 2 случая:

1. Если $n < k$, то надо приписать в конец исходной строки $k - n$ минимальных символов из s.

2. Если $n \geqslant k$, то надо суффикс строки, состоящий из первых k символов исходной строки, из самых больших символов заменить на самые минимальные символы, а следующий символ после этого суффикса увеличить до следующего в алфавите символа, присутствующего в исходной строке.

Асимптотика $O(n + k)$.

Разбор задачи «Умный обогреватель»

Заметим, что ограничения на t_{min} и t_{max} задают только последовательности работы обогревателя равные `00001`, `00000`, `11110`, `11111`.

1. `00000` на позициях $[i; i+4]$ значит, что $t_{min} \leqslant \max\limits_{i \leqslant j \leqslant i+4} a_j$.

2. `00001` на позициях $[i; i+4]$ значит, что $t_{min} \geqslant \max\limits_{i \leqslant j \leqslant i+4} a_j + 1$.

3. `11111` на позициях $[i; i+4]$ значит, что $t_{max} \geqslant \min\limits_{i \leqslant j \leqslant i+4} a_j$.

4. `11110` на позициях $[i; i+4]$ значит, что $t_{max} \leqslant \min\limits_{i \leqslant j \leqslant i+4} a_j - 1$.

В итоге, мы получаем набор ограничений на верхнюю и нижнюю границу для t_{min} и t_{max}. Поскольку гарантируется, что ответ существует, то есть всего два случая:

1. Отрезки пересекаются, тогда ответ — $t_{min} = t_{max} =$ любое число из пересечения.

2. Отрезки не пересекаются, тогда надо выбрать максимально возможное значение t_{min} и минимально возможное значение t_{max}.

Асимптотика $O(n)$.

Разбор задачи «Оптимизация акции»

Сначала решим задачу за $O(n^2 \log n)$. Это можно сделать при помощи динамики dp_i — минимальная стоимость разбиения префикса массива длины i. Пересчет $dp_0 = 0$, $dp_i = \min\limits_{j<i} dp_j + cost_{j+1,i}$, где $cost_{l,r}$ — сумма $(r-l+1) - \lfloor \frac{r-l+1}{10} \rfloor$ максимумов на подотрезке $[l;r]$. В момент пересчета мы можем идти по j от $i-1$ до 0, поддерживая текущие элементы массива на подотрезке $[j+1;i]$ в $std::multiset$ и аккуратно пересчитывая сумму $\lfloor \frac{r-l+1}{10} \rfloor$ минимумов.

Чтобы решить задачу быстрее, надо понять, что нам всегда выгодно брать отрезки либо длины 1, либо длины 10. Предположим мы взяли отрезок длины меньшей 10, тогда его стоимость не зависит от разбиения и его можно разбить на отрезки длины 1, предположим мы взяли отрезок длины x, $10 \leqslant x \leqslant 19$, тогда мы можем оставить из этого отрезка оптимальный подорезок длины 10, а все что по краям взять отрезками длины 1. Если длина отрезка равна 20, то, очевидно, не хуже взять его как два отрезка длины 10. В остальных случаях можно также предъявить разбиение на подотрезки длины 1 и 10, которое будет не дороже исходного.

Делать пересчет для отрезков длины 1 легко, чтобы делать пересчеты для отрезка длины 10, можно хранить элементы из отрезка $[i-9;i]$ в любой структуре данных, умеющей брать минимум быстро. Например очередь с минимумом или $std::multiset$.

Асимптотика $O(n)$.

Разбор задачи «Лабиринт»

Назовём "ёжиком"на n вершинах дерево, образованное подвешиванием n листьев к $n+1$-й вершине, которую назовём корнем ёжика.

Для решения задачи необходимо было рассмотреть три случая:

1. Ответ на задачу является ёжиком. Несложно заметить, что в ёжике на n вершинах $\frac{n(n-1)}{2}$ диаметров.

2. Ответ на задачу — это два ёжика, чьи корни соединены ребром. Если в первом ёжике p вершин, а во втором q вершин, то в получившемся дереве $p \cdot q$ диаметров.

3. Ответ на задачу — это несколько ёжиков, чьи корни подвесили к новой вершине, которая будет корнем нового дерева. Пусть размеры ёжиков равны $s_1, s_2, s_3, \ldots, s_k$. Тогда количество диаметров будет равно $\frac{(s_1+s_2+s_3+\ldots s_k)^2 - s_1^2 - s_2^2 - s_3^2 - \ldots - s_k^2}{2}$. Для того, чтобы найти минимальное суммарное число вершин в ёжиках, для которых значение описанной выше функции будет равно n, можно воспользоваться методом динамического программирования. Пусть $dp_{i,j}$ — минимально возможное количество вершин в дереве, сумма размеров ёжиков которого равна i, а количество диаметров равно j. Для перехода достаточно перебрать размер следующего ёжика.

Данное решение работает за $O(n^3)$, однако его можно прооптимизировать до $O(n^2)$. Оставим эту оптимизацию в качестве упражнения.

Отметим, что при ограничениях 500 было также возможно использовать перебор с отсечениями вместо динамического программирования.

Разбор задачи «Машинное обучение»

Сначала оценим минимальную длину отрезка, чтобы ответ был c, для этого на этом отрезке должно быть какое-то число в единственном экземпляре, какое-то в двух и.т.д. Тогда длина отрезка равна $1 + 2 + 3 + \cdots + c = \frac{c \cdot (c+1)}{2} \Rightarrow$ ответ никогда не будет превосходить $2 \cdot \sqrt{n}$ для любого запроса.

Теперь сопоставим числам из массива и запросов числа от 1 до $n+q$ так, чтобы равным числам соответствовали равные числа, а не равным разные. Очевидно, что такое преобразование над массивом не влияет на ответ, но теперь все числа не превосходят $n+q$.

Пусть мы для отрезка $[l;r]$ знаем количество каждого числа на этом отрезке cnt_i, и количество каждого количества $size_i$. Тогда мы можем за $O(\sqrt{n})$ найти Мех этого множества найдя первый 0 в

массиве $size_i$. Также мы умеем за $O(1)$ найти пересчитать массивы cnt_i и $size_i$ для отрезков $[l-1;r]$, $[l+1;r]$, $[l;r-1]$, $[l;r+1]$.

Теперь будем представлять наш отрезок как тройку чисел (t,l,r), где t — количество изменений в хронологическом порядке по запросам и соответствующий подотрезок $[l;r]$. Лекго видеть, что t тоже можно двигать на единицу за $O(1)$. Заметим, что $q \approx n$ и заменим q на n в последующий оценках.

Возьмем константу $P = n^{\frac{2}{3}}$ и отсортируем наши запросы по тройкам $(\lfloor \frac{t}{P} \rfloor, \lfloor \frac{l}{P} \rfloor, r)$ и будем обрабатывать запросы в таком порядке переводя предыдущий отрезок в следующий. Граница r будем увеличиваться на $O(n)$ на каждый квадрат размера $\frac{n^2}{P^2} = \frac{n^2}{n^{\frac{4}{3}}} = n^{\frac{2}{3}} \Rightarrow$ суммарно перемещений правой границы $O(n^{\frac{5}{3}})$. Также на каждый запрос каждая из границ t и l двигается не более чем на $O(n^{\frac{2}{3}})$ $\Rightarrow$ суммарно перемещений этих границ $O(n^{\frac{5}{3}})$.

Асимптотика решения $O(n^{\frac{5}{3}})$.