Задача A. Среднее

Нам дана последовательность B, где B_i — это среднее арифметическое первых i чисел последовательности A:

$$B_i = \frac{A_1 + A_2 + \cdots + A_i}{i}$$

Введём префиксные суммы: $S_i = A_1 + A_2 + \cdots + A_i$. Тогда по определению среднего имеем $B_i = \frac{S_i}{i}$, откуда $S_i = i \cdot B_i$. Сразу заметим, что последовательность S можно явно посчитать.

Из определения последовательности S:

$$S_i = S_{i-1} + A_i$$

значит

$$A_i = S_i - S_{i-1}$$

Таким образом, для $i > 1$, мы умеем вычислять A_i с помощью формулы выше. Как же вычислить A_1? несложно заметить, что исходя из формулы последовательности B: $B_1 = \frac{A_1}{1} = A_1$, тогда $A_1 = B_1$.

Задача B. Новогодние огни

Разберем некоторые подзадачи, которые были подсказками для полного решения:

Подзадача 1. Если $N \geqslant 1$, ставим фонарь ровно в точку ёлки p_1 (это разрешено), расстояние до фонаря 0, до любого снеговика > 0, значит ёлка точно достанется Деду. Ответ $= t_1$.

Подзадача 2. Один фонарь. Достаточно рассмотреть каждый интервал между соседними снеговиками отдельно: в крайних интервалах можно забрать всю сумму, а во внутреннем интервале максимум равен «максимальной сумме яркостей в скользящем окне фиксированной длины» — подробнее в полном решении.

Подзадача 3. Снеговиков нет, конкурента нет: все ёлки достаются Деду. Ответ $\sum_{i=1}^{K} t_i$.

Подзадача 4. Случай $M = 0$ уже рассмотрен. Если $M = 1$, есть два крайних интервала $(-\infty, f_1)$ и $(f_1, +\infty)$, каждый целиком берётся одним фонарём. Считаем суммы яркостей слева и справа от f_1 и берём сумму лучших N из этих двух чисел.

Рассмотрим **полное решение.** Остальные подзадачи решались, если придумать часть полного решения и/или реализовать его недостаточно эффективно.

Напомним, что во входных данных не гарантируется, что координаты отсортированы по возрастанию, поэтому это необходимо сделать — получим отсортированные координаты

$$f_1 < f_2 < \cdots < f_M$$

Они разбивают прямую на $M + 1$ интервалов: $(-\infty, f_1)$, (f_1, f_2), $\ldots$, (f_{M-1}, f_M), $(f_M, +\infty)$.

Для каждого интервала будем обозначать через S сумму яркостей всех ёлок этого интервала.

Заметим, что в интервалах $(-\infty, f_1)$ и $(f_M, +\infty)$ **достаточно одного фонаря** (можно поставить его сразу перед f_1 / после f_M), чтобы забрать все ёлки интервала. Таким образом, больше одного фонаря ставить в эти интервалы нет смысла.

Посмотрим, что происходит во внутренних интервалах.

Рассмотрим внутренний интервал $(L, R) = (f_j, f_{j+1})$. Двух фонарей всегда хватает, чтобы забрать все ёлки интервала (один ставим очень близко справа от L, второй — очень близко слева от R), значит максимум на 2 фонаря равен S.

Осталось найти максимум на 1 фонарь. Поставим фонарь в точку $q \in (L, R)$ и посмотрим, какие ёлки он «захватывает»:

Ёлка в точке p достаётся Деду, если расстояние до фонаря меньше расстояния до ближайшего снеговика. Во внутреннем интервале ближайший снеговик к p — это L или R, и условие превращается в две строгие неравенства:

$$|p - q| < p - L \quad \text{и} \quad |p - q| < R - p$$

Получаем, что p должен лежать в интервале

$$(\tfrac{L+q}{2}, \tfrac{R+q}{2})$$

Важное замечание: длина этого интервала не зависит от q и равна

$$\frac{R+q}{2} - \frac{L+q}{2} = \frac{R-L}{2}$$

Пусть мы выбрали некоторый набор ёлок внутри интервала, и среди них минимальная позиция p_l, максимальная позиция p_r. Тогда существует положение одного фонаря, которое заберёт все эти ёлки, тогда и только тогда, когда

$$2 \cdot (p_r - p_l) < (R - L)$$

Обозначим через A максимум яркости, который можно забрать в интервале (L, R) **одним** фонарём. Давайте научимся считать эту величину.

Рассматриваем те ёлки, чьи p_i попадают внутрь интервала (L, R).

Для фиксированного внутреннего интервала (L, R) пусть его ёлки идут по возрастанию p. Держим два указателя l и r и текущую сумму яркостей cur в окне $[l..r]$:

- увеличиваем r слева направо, прибавляя t_r к cur;
- пока условие нарушено, то есть $2 \cdot (p_r - p_l) \geqslant (R - L)$, двигаем l вправо и вычитаем t_l из cur;
- обновляем $A = \max(A, cur)$.

Итак, для внутреннего интервала у нас есть:

- A — максимум на 1 фонарь,
- S — максимум на 2 фонаря.

Осталось научиться комбинировать различные интервалы между собой. Для каждого интервала у нас есть a_i, s_i — сколько мы получим, если поместим в интервал 1 и 2 фонаря соотвественно. Давайте считать «профит», который нам приносит каждый фонарь: первый фонарь интервала принесет нам a_i яркости, второй: $s_i - a_i$ яркости. Тогда, давайте набирать фонари жадно — каждый раз брать тот фонарь, который приносит нам больше яркости. Несложно показать, что это будет оптимальная стратегия.

Получается, что для решения задачи нужно было:

1. каждый интервал разбить на 1 или 2 оптимальных фонаря
2. узнать, сколько яркости дает нам очередной фонарь, посчитав A и S
3. взять самые большие N значений среди полученных яркостей.

С учетом сортировки, решение будет работать за $\mathcal{O}(K \log K + M \log M)$.

Задача C. Новогодняя MST

Первое наблюдение: если $a < b$, то $a \bmod b = a$, значит $w(a, b) = \min(a, b \bmod a) = b \bmod a$. То есть для $a < b$ всегда $w(a, b) = b \bmod a$.

Подзадача 1. Если $N \leqslant 1000$, можно построить граф явно: в нем получится $\mathcal{O}(n^2)$ ребер, и на этом графе можно построить MST любым удобным способом — например, алгоритмом Краскала, Прима или Борувки. Асимптотика такого решения будет $\mathcal{O}(n^2 \log n)$, что вполне приемлемо при данных ограничениях.

Подзадача 2. Заметим, что из ограничения следует, что различных p_i может быть не более 1000. А что будет, если $p_i = p_j$ при $i \neq j$? Заметим, что такие вершины мы можем соединить ребром веса 0, и «сжать» их в одну вершину. Таким образом, в графе можно оставить только различные p_i, и в нем будет достаточно мало вершин, и можно будет воспользоваться решением из первой подзадачи.

Подзадача 3. Аналогично второй подзадаче, сожмем различные веса в один. Тогда останется $n \leqslant 3 \cdot 10^4$ вершин. Текущий граф — полный, значит, на нем можно воспользоваться алгоритмом

Прима для полных графов, что будет работать за $\mathcal{O}(n^2 + m)$, где m будет порядка $5 \cdot 10^8$. Если не хранить граф явно, это легко вписывается в ограничения задачи по времени и по памяти.

Для полного решения аналогично сожмем различные p_i в одну. Положим как M максимальное из всех p_i.

Для каждого значения p_x мы переберём все его кратные: $k \cdot p_x \leqslant M$, то есть $k = 1, 2, \ldots, \left\lfloor \frac{M}{p_x} \right\rfloor$. Для каждого такого кратного $k \cdot p_x$ мы хотим найти вершину y со значением p_y, которая:

- является минимальной среди всех вершин, для которых $p_y \geqslant k \cdot p_x$;
- а в случае $k = 1$ берём минимальную вершину со строго большим значением, то есть $p_y > p_x$ (чтобы не выбрать саму вершину x).

Найдя такую вершину y, мы добавляем ребро (x, y) в список рёбер, которые будем обрабатывать (то есть потенциально использовать в MST), и называем его *хорошим*.

Лемма 1: существует MST только из хороших ребер.

Сначала научимся решать задачу, используя **Лемму 1**, а потом докажем её.

Ответим на вопрос — сколько *хороших* ребер существует? По построению, для вершины с весом x мы добавили не более $\frac{M}{x}$ ребер. Таким образом, суммарно мы добавили не более

$$\frac{M}{1} + \frac{M}{2} + \cdots + \frac{M}{M} = \mathcal{O}(M \log M)$$

ребер. Таким образом, в *хорошем* графе $\mathcal{O}(n)$ вершин и $\mathcal{O}(M \log M)$ ребер. Тогда, наивно написанный алгоритм Краскала/Прима отработает на этом графе за $\mathcal{O}(M \log^2 M)$, что пройдет **четвертую подзадачу.**

Пятную подзадачу можно было решить таким образом: заметим, что в алгоритме Краскала самая долгая часть — сортировка ребер по весам. Заметим, что веса $\leqslant M$, тогда поймем, что можно было воспользоваться сортировкой подсчетом, и получить асимптотику $\mathcal{O}(M \log M \alpha(n))$, что легко укладывается в ограничения. Также существовали альтернативные (и достаточно быстрые) решения, использующие алгоритм Борувки.

Доказательство Леммы 1.

Предположим противное: не существует MST, состоящего только из хороших рёбер. Тогда возьмем любое MST — тогда в нем есть не *хорошее* ребро $a - b$. Пусть $p_a < p_b$, и пусть k таково, что $k \cdot p_a \leqslant p_b < (k+1) \cdot p_a$. Тогда, поскольку мы не выделили ребро (a, b), существуют вершины $c_1, c_2, \ldots, c_n$ такие, что выполняется $k \cdot p_a \leqslant p_{c_1} < p_{c_2} < \cdots < p_{c_n} < p_b$, а также (в случае $k = 1$) дополнительно требуется $p_a < p_{c_1}$. Но тогда мы выделили рёбра (a, c_1), (c_1, c_2), (c_2, c_3), $\ldots$, (c_{n-1}, c_n), (c_n, b). Их суммарная стоимость равна $p_b - k \cdot p_a = p_b \bmod p_a$. Следовательно, мы можем удалить ребро (a, b) и вместо него поставить путь из a в b (целиком или частично), и при этом граф останется связным, а суммарный вес станет меньше или равен прежнему.

Получили противоречие — значит, существует MST только из хороших ребер, и MST, которое мы найдем, будет минимальным и во всем графе.

Задача D. Новогодние билетики

Рассмотрим бинарную операцию op, которая будет вычислена в последнюю очередь. Тогда все выражение будет равно $op(expr_1, expr_2)$, где $expr_1$ — значение выражения слева от операнда, а $expr_2$ — справа. Таким образом, для каждой строки из цифр s можно найти множество значений $vals[s]$, которые могут получиться, если расставить в этой строке операции и скобки. Это может быть либо просто число s, либо $op(a, b)$, где op — любая операция, $a \in vals[s_1]$, $b \in vals[s_2]$, $s = s_1 s_2$.

Несложно заметить, что $|vals[s]| \leqslant 2 \cdot 10^{len(s)}$, потому что для любой операции $len(op(a, b)) \leqslant len(a) + len(b)$, где $len(x)$ — количество десятичных цифр в числе x.

Также, заметим, что для строк длины 6 не нужно искать всё множество $vals$, а достаточно только проверить, принадлежит ли ему 100. Аналогично, для множеств для строк длины 5 не нужно находить множество целиком, достаточно найти только значения, которые при применении операции с числом длины 1 могут дать 100.

Также, нужно не забыть про операцию отрицания.

Такое решение укладывается в ограничения и находит все решения.