Задача A. Ділення націло на 5

Виведіть $min(n \bmod 5, 5 - n \bmod 5)$.

Задача B. Медіанне божевілля

Спитаємо спершу запит про всі елементи крім якогось одного. Зрозуміло, що нам повернуть x, позицію числа $\frac{n}{2}$ чи $\frac{n}{2}+1$. Тепер, якщо ми спитаємо запит про всі числа крім x, ми отримаємо y, позицію другого з цих чисел.

Помітимо, що наші запити не дозволяють відрізнити перестановку $(p_1, p_2, \ldots, p_n)$ від перестановки $(n+1-p_1, n+1-p_2, \ldots, n+1-p_n)$. Тому вважатимемо, що $p_x = \frac{n}{2}$, $p_y = \frac{n}{2}+1$, і знайдемо перестановку за цієї умови, а в кінці замінимо її на $(n+1-p_1, n+1-p_2, \ldots, n+1-p_n)$, якщо виявиться що $p_1 > p_n$.

Нехай на якомусь кроці ми знаємо позиції всіх чисел від t до $n+1-t$. Тоді виберемо якесь pos, число в якому ми ще не знаємо, і спитаємо запит про всі позиції, числа в яких ми не знаємо, крім pos. Ми отримаємо x_1 — позицію числа $t-1$ чи $n+2-t$. Спитаємо тепер про всі позиції, числа в яких ми не знаємо, крім x_1, отримаємо y_1, позицію другого з цих чисел. Тепер нам треба зрозуміти, хто з p_{x_1} та p_{y_1} $t-1$, а хто $n+2-t$.

Для цього спитаємо запит з x_1, x, y. Якщо позицією медіани виявиться x, то $x_1 = t-1$, інакше $x_1 = n+2-t$.

Таким чином, ми можемо знаходити позиції двох наступних чисел за 3 операції.

Задача C. Абсолютно неадекватні операції

Розглянемо префіксні суми $pref_1, pref_2, \ldots, pref_n$ ($pref_i = a_1 + a_2 + \ldots + a_i$). Легко помітити, що коли ми робимо операцію з позицією i, то префіксні суми $pref_{i-1}$ та $pref_i$ міняються місцями, а інші не змінюються.

Зрозуміло, що є однозначна відповідність між різними масивами префіксних сум та різними масивами, отже досить порахувати кількість різних масивів префіксних сум. Наша операція дозволяє поміняти місцями довільні дві сусідні з перших $n-1$ сум, а отже ми можемо отримати будь-яку перестановку цих сум.

Кількість конфігурацій тоді рівна $\frac{(n-1)!}{\prod_x cnt_x!}$, де cnt_x позначає скільки разів дане x зустрічається серед перших $n-1$ префіксних сум.

Задача D. Паліндромна лихоманка

Достатньо вивести $t = reverse(s)$ (під $reverse(s)$ мається на увазі перевернутий рядок s. Наприклад, для $s = abc$, ми можемо вибрати $t = cba$.

Дійсно, $reverse(s + reverse(s)) = s + reverse(s)$, і $reverse(reverse(s) + s) = reverse(s) + s$, тому ці рядки дійсно є паліндромами.

Задача E. Шлях додому

Нехай P_i — позиція i-го міста. Нехай міста в маршруті — $i_1, i_2, \ldots, i_k$. Тоді сумарна довжина маршруту — $\sum_{1 \leqslant i \leqslant k-1} |P_{i+1} - P_i|$.

Отже, вклад міста визначається лише тим, по яку сторону воно знаходиться від своїх сусідів. Будемо додавати міста в порядку зростання, і тримати в dp кількість міст, які ми вибрали на цю мить, скільки шматків маршруту в нас є на цю мить, сумарний вклад міст до цього моменту по модулю m, а також чи додали ми вже початок та кінець всього маршруту. При обробці чергового міста перебирайте всі варіанти: місто до нього в маршруті знаходиться до нього/після нього/воно само є лівим кінцем маршруту, і так само для правого сусіда в маршруті.

Обмеження в задачі достатньо малі, щоб таке рішення за $O(nk^2m)$ з великою константою проходило в TL.

Задача F. ПерестановочкА

Помітимо спершу, що задача, по суті, просить порахувати нас кількість таких відрізків (i, j), що p_i та p_j це максимальний та мінімальний елемент серед чисел $p_i, p_{i+1}, \ldots, p_j$ (в якомусь порядку). Дійсно: у всіх таких відрізків пари (min, max) різні, а для відрізку (i, j), для якого умова вище не виконується, пара (min, max) співпадає з парою (i_1, j_1), де i_1, j_1 — позиції мінімуму та максимуму на цьому відрізку.

Будемо називати такі відрізки хорошими. Очевидно, всі відрізки (i, i) хороші.

Порахуємо спершу кількість відрізків (i, j) з $i < j$, для яких p_i мінімальне серед $p_i, p_{i+1}, \ldots, p_j$, а p_j максимальне, а потім агалогічно порахуємо ті, для яких p_i максимальне і p_j мінімальне.

Для початку для кожного i порахуємо мінімальне lim_i, що p_i — максимальне серед $p[lim_i : i]$. Це класична задача, і це можна зробити одним проходом зі стеком.

Тепер будемо йти зліва направо і тримати в стеці лише такі позиції x, що справа від них (до моменту, який ми розглядаємо наразі) немає елементів, менших за них. Нехай зараз ми обробляємо елемент p_i. Спершу ми видаляємо зі стеку останню позицію x, поки $p_x > p_i$. Тепер, для кожного x, що лишився в стеку, p_x є мінімумом на $p[x : i]$, і всі такі x, для яких p_x є мінімумом на $p[x : i]$, лежать в стеці. Нам лишилось знайти, для скількох з цих x p_j є максимумом на $p[x : j]$.

Це рівносильно тому, що ми хочемо знайти кількість позицій в стеці, що не менші за lim_i. Це просто зробити бінарним пошуком (за умови, якщо ми пишемо стек на масиві), адже позиції в стеку відсортовані.

Загальна асимптотика $O(n \log n)$.

Задача G. ПерестановочкИ

Помітимо спершу, що задача, по суті, просить порахувати нас кількість таких відрізків (i, j), що p_i та p_j це максимальний та мінімальний елемент серед чисел $p_i, p_{i+1}, \ldots, p_j$ (в якомусь порядку). Дійсно: у всіх таких відрізків пари (min, max) різні, а для відрізку (i, j), для якого умова вище не виконується, пара (min, max) співпадає з парою (i_1, j_1), де i_1, j_1 — позиції мінімуму та максимуму на цьому відрізку.

Далі зауважимо, що достатньо порахувати математичне очікування $f(p)$ по всім перестановкам, і помножити його на $n!$.

Знайдемо для кожної пари (i, j) імовірність того, що p_i та p_j це максимальний та мінімальний елемент серед чисел $p_i, p_{i+1}, \ldots, p_j$ (в якомусь порядку). Для $i = j$ ця імовірність рівна 1, тому ми додаємо до математичного очікування n. Для $i < j$ ця імовірність рівна $\frac{2}{(j-i+1)(j-i)}$, адже це просто ймовірність того, що мінімальний елемент з цього відрізку потрапив в один з його кінців ($\frac{2}{j-i+1}$), а потім максимальний потрапив в інший кінець ($\frac{1}{j-i}$).

Таким чином, тепер для кожного відрізка довжини len потрібно додати до відповіді $\frac{2}{len(len-1)}$. Отже, математичне очікування рівне

$$n + \sum_{2 \leqslant len \leqslant n} (n + 1 - len) \frac{2}{len(len-1)}$$

. Щоб отримати відповідь, достатньо помножити це на $n!$.

Задача H. Вишуканий максимум

Відсортуємо всі числа в порядку зростання.

Помітимо, що $\frac{a_i a_j}{|a_i - a_j|} = \frac{1}{|\frac{1}{a_i} - \frac{1}{a_j}|}$. Якщо ми хочемо максимізувати $\frac{1}{|\frac{1}{a_i} - \frac{1}{a_j}|}$, то це означає, що ми хочемо мінімізувати $|\frac{1}{a_i} - \frac{1}{a_j}|$, а отже достатньо розглядати лише пари сусідніх чисел.

Таким чином алгоритм виглядає наступним чином: відсортуємо всі числа, і знайдемо максимальне з чисел $\frac{a_i a_{i+1}}{|a_i - a_{i+1}|}$

Задача I. Надзвичайно оригінальна задача про кістякове дерево

Коли у нас є задача, де потрібно знайти мінімальне кістякове дерево і ребра задані неявно, то часто буває корисно подумати в сторону алгоритму Борувки.

Далі йде приблизний опис алгоритму Борувки (для кращого розуміння погугліть):

У будь-який момент часу алгоритму наші вершини будуть розбиті на множини, причому вершини з однієї множини вже пов'язані між собою, а вершини з різних ні. Далі, ми повинні якимось чином знайти вагу мінімального ребра, що виходить з кожного числа й допровести ці ребра. Кількість множин зменшиться як мінімум в 2 рази, тобто кількість ітерацій нашого алгоритму не більш логарифмічна від числа вершин.

Таким чином, все що нам залишилося, це вміти знаходити вагу мінімального ребра, що виходить з множини. Для цього, для кожного числа i знайдемо інше число j, таке що $(a_i+a_j) \bmod M$ мінімально і числа i, j належать різним множинам. Це можна робити зберігаючи всі числа в сеті, потім видаляти всі числа з однієї множини, знаходити для кожного з них оптимальне j (за допомогою lower bound), повертати ці числа назад.

Задача J. Коли немає чим зайнятись, а вдома лише перестановка

Порахуємо спершу, скільки секунд займе ітерація для перестановки вигляду $(k, 1, 2, \ldots, k-1, k+1, \ldots, n)$. При $k > 1$ попередня перестановка до неї — $(k-1, n, n-1, \ldots, k, k-2, \ldots, 2, 1)$. Можна помітити, що в цій перестановці не змінився порядок лише таких пар елементів: $(k-1, k+1), (k-1, k+2), \ldots, (k-1, n)$, а також $(k, 1), (k, 2), \ldots, (k, k-2)$. Всього не змінився порядок рівно в $(n-k)+(k-2)=(n-2)$ пар, а отже для кожної з інших $\frac{n(n-1)}{2}-(n-2)$ пар буде операція, в якій ми міняємо місцями числа з цієї пари. Таким чином, ця ітерація займає $\frac{n(n-1)}{2}-(n-2)=cnt_n$

Порахуємо спершу dp_n — за скільки секунд перестановка $(n, n-1, n-2, \ldots, 1)$ перетвориться на перестановку $(1, 2, 3, \ldots, n)$. Спершу ми витратимо dp_{n-1} секунд, щоб отримати $(n, 1, 2, \ldots, n-1)$, а далі ми $n-1$ разів будемо виконувати перехід з першого абзацу (витрачаючи по cnt_n секунд), і $n-1$ разів сортувати останні $n-1$ елемент. Всього маємо $dp_n=(n-1)cnt_n+ndp_{n-1}$.

Тепер будемо йти справа наліво. Нехай на цю мить в нас відсортовані елементи від p_{n-x+1} до p_n в порядку зростання. Тоді, якщо p_{n-x} більше за рівно t чисел з p_{n-x+1} до p_n, то за $t(cnt_{x+1}+dp_x)$ стануть відсортованими і числа від p_{n-x} до p_n. Порахувати це t можна з допомогою дерева відрізків/Фенвіка/ordered_set.

Просумуємо це по всім x і отримаємо відповідь.

Задача K. Чергове розчарування: задача на парування

Покажемо, що оптимальне парування для даного масиву — це об'єднати в пари $(1, 1 + \frac{n+1}{2}), (2, 2 + \frac{n+1}{2}), \ldots, (1 + \frac{n+1}{2}, n + 1)$. Будемо позначати парування як $\frac{n+1}{2}$ пар (l_i, r_i), де $1 \leqslant l_i < r_i \leqslant n + 1$, і кожне число від 1 до $n + 1$ зустрічається рівно один раз як l_i чи r_i.

Для цього розглянемо парування з найбільшою сумарною вагою ребер, серед них — з найбільшою сумою $|r_i - l_i|$, а серед них — з найменшою сумою $|r_i - l_i|^2$.

Якщо є дві пари (l_i, r_i) і (l_j, r_j), що не перетинаються, тобто $l_i < r_i < l_j < r_j$, то вигідно замінити їх на (l_i, l_j), (r_i, r_j). Сумарна вага ребер не зменшиться, а сума $|r_i - l_i|$ збільшиться.

Якщо є дві пари (l_i, r_i) і (l_j, r_j), одна з яких містить іншу, тобто $l_i < l_j < r_j < r_i$, то вигідно замінити їх на (l_i, r_j), (l_j, r_i). Можна показати, що сумарна вага ребер не зменшиться, сума $|r_i - l_i|$ не зміниться, а сума $|r_i - l_i|^2$ зменшиться.

Таким чином, маємо що в такому паруванні кожні два ребра перетинаються, але не містять одне одного. Єдине таке парування приведено в першому реченні.

Лишилось знайти суму $\sum_{1 \leqslant t \leqslant \frac{n+1}{2}} max(a_t, a_{t+1}, \ldots, a_{t+\frac{n+1}{2}-1})$. по всім перестановкам даного масиву. Для цього досить знайти маточікування $\sum_{1 \leqslant t \leqslant \frac{n+1}{2}} max(a_t, a_{t+1}, \ldots, a_{t+\frac{n+1}{2}-1})$ по випадковій перестановці даного масиву. Очевидно, що маточікування цього max однакове для всіх t, тому лишилось знайти маточікування максимуму випадково вибраної підмножини з $x = \frac{n+1}{2}$ елементів. Це вже зробити простіше: для i-го елементу в порядку зростання ймовірність того, що він буде максимумом, рівна $\frac{C(i-1,t-1)}{C(n,t)}$, а отже вклад в маточікування рівний $a_i \frac{C(i-1,t-1)}{C(n,t)}$.

Задача L. Задача на кореневу декомпозицію з великими обмеженнями

Помітимо, що $n \bmod (n-x) = ((n-x)+x) \bmod (n-x) = x \bmod (n-x) = x$ при $x < \frac{n}{2}$. Отже, всі числа що строго менші за $\frac{n}{2}$ зустрінуться серед даних чисел.

З іншого боку, $n \bmod (n-x) = ((n-x)+x) \bmod (n-x) = x \bmod (n-x) \leqslant min(x, n-x-1) \leqslant \frac{x+(n-x-1)}{2} = \frac{n-1}{2}$. Отже, всі числа з умови строго менші за $\frac{n}{2}$.

Таким чином, числа з умови — це цілі числа від 0 до $\lfloor \frac{n-1}{2} \rfloor$, їх рівно $\lfloor \frac{n+1}{2} \rfloor$. Отже, достатньо просто вивести $\lfloor \frac{n+1}{2} \rfloor$!

Примітка: Замість викладок вище можна просто подивитись на відповіді для маленьких n та помітити закономірність.