Задача A. Магазини

Якщо ми додамо кількість магазинів на першому і другому шляху $a + b$, ми порахуємо деякі магазини двічі. А саме, ті, які є на обох шляхах. А таких магазинів c. Тому віднімемо цю кількість — відповідь $a + b - c$.

Задача B. Складаємо масив

Відповідь — масив відсортований за незростанням.

Чому так? Порахуймо суму всіх конкатенацій $\sum_{i=1}^{n}\sum_{j=i}^{n} \text{concat}(a_i, a_j) = \sum_{i=1}^{n}\sum_{j=i}^{n}(10 \cdot a_i + a_j) = \sum_{i=1}^{n}(10 \cdot a_i \cdot (n-i+1) + \sum_{j=i}^{n} a_j) = \sum_{i=1}^{n} 10 \cdot a_i \cdot (n-i+1) + \sum_{i=1}^{n}\sum_{j=i}^{n} a_j = \sum_{i=1}^{n} 10 \cdot a_i \cdot (n-i+1) + \sum_{i=1}^{n} i \cdot a_i = \sum_{i=1}^{n} a_i \cdot (10 \cdot (n - i + 1) + i) = \sum_{i=1}^{n} a_i \cdot (10n - 10i + 10 + i) = \sum_{i=1}^{n} a_i \cdot (10n - 9i + 10) = \sum_{i=1}^{n} a_i \cdot (10n + 10) - \sum_{i=1}^{n} a_i \cdot 9i$

Перша частина не змінюється від перестановки масиву a, а щоб мінімізувати значення другої суми, треба відсортувати масив за незростанням. Доводиться просто — якщо є два елементи a_i, a_j, що $a_i < a_j$ та $i < j$, то їх можна поміняти місцями й відповідь не збільшиться.

Задача C. Всі полюбляють перестановки

Кожну перестановку можна розбити на "цикли".

Позначимо $p^k(i) = p[p^{k-1}(i)]$, і $p^0(i) = i$. Будемо перебирати k за зростанням з 1. Колись ми зациклимось, тому що прийдемо в якийсь елемент, де вже були.

1. Чому це колись станеться? Якби ми не знайшли такий цикл, то це означало що в нас є нескінченно багато елементів в перестановці.

2. Чому це буде саме цикл? Якби це був не цикл, то для якогось числа x його число входжень у перестановку було більше за 1.

Ми розбили перестановку на цикли. Тепер треба порахувати суму індексів. Розвернемо цикл в іншу сторону, щоб йти так, як нам сказано в задачі. Тоді, нам залишається тільки зробити префіксні суми на цих циклах і акуратно підрахувати відповідь — порахувати скільки разів цикл повність пройдеться, і скільки ще елементів після повного проходу воно обійде.

Асимптотика — $\mathcal{O}(N)$

Задача D. Кріт

Порахуймо для кожної клітинки максимальну відстань від неї до дірки. Наївне розв'язання працюватиме за $\mathcal{O}((nm)^2)$. Зробімо трошки розумніше. Будемо запускати алгоритм бфс з дірок, і після того, як він знайде мінімальну відстань від дірки до всіх клітин, оновимо масив з максимальними відстанями.

Далі знаходимо мінімум в масиві максимальних відстаней і виводимо всі клітинки, які мають таку максимальну відстань. Таким чином отримуємо асимптотику $\mathcal{O}(nmk)$, де k — кількість дірок.

Планувалось дати ще блок, де гарантується, що між кожними двома клітинами, що не містять квіт, існує не більше ніж 1 шлях, і нема обмеження на кількість дірок. Але, він не пройшов у фінальний етап, бо був окремо складніший за всі інші блоки.

Задача E. НСД, Сума, Помножити. що?...

Перший важливий факт для розв'язання цієї задачі — gcd змінюється не більше, ніж $\log A$ разів. Це можна просто показати, нехай в нас g_1, g_2, де $g_2 = gcd(g_1, x)$. Тепер, якщо $g_1 \neq g_2$, то $\frac{g_1}{g_2} > 1$ (бо g_2 ділить g_1), тобто хоча б 2, що значить, що кожен раз коли gcd змінюється, воно зменшується хоча б вдвічі.

Тепер, давайте для кожної позиції i знайдемо позиції j зліва і k справа, де gcd змінюється. Але, треба запам'ятати для кожного gcd останню позицію, так, щоб довжина відрізка $[j; i]$ чи $[i; k]$ була максимальною. Для кожної позиції в нас є $\mathcal{O}(\log A)$ позицій, де gcd змінюється.

Зафіксуймо праву границю r, тоді щоб відповідати на запити підтримуймо масив b, що b_i — найбільша відповідь для відрізків, що мають ліву границю в i, і їх права границю менше рівна r.

Щоб відповісти на запит $[l;r]$ питаємо максимум в масиві b на відрізку $[l;r]$ і треба перевірити всі відрізки, які починались в цьому відрізку, але закінчились лівіше. Тоді a_l точно було взято у відрізок, то перевіримо всі відрізки, де змінюється gcd справа від l.

Таке можна розв'язувати офлайн за допомогою дерева відрізків і сортуванням запитів, чи онлайн, за допомогою персистентного дерева відрізків.

Розв'язок з персистентним деревом відрізків має асимптотику $\mathcal{O}(N \log N \log N + Q \cdot (\log A + \log N))$. Інші розв'язки, які відомі мені, мають таку саму асимптотику, але кращу константу, що робить їх значно швидшими.

На початку планувалось зробити всі запити онлайн щоб було цікавіше розв'язувати задачу, але через деякі причини було вирішено зробити задачу стандартною — хто хоче розв'язувати офлайн — нехай розв'язує

Автор усіх задач: Андрій Столітній

Задача A. Заявки

Щоб було n сторінок з вакансіями, треба, щоб на $n-1$ з них було по 20 вакансій, і ще хоча б одна, щоб створити нову сторінку — $20 \cdot (n-1)+1$. Щоб не створити $n+1$-у сторінку, треба, щоб було менше ніж $20 \cdot n + 1$ вакансій, тобто максимальна кількість — $20 \cdot n$.

Задача B. Хрестики-нулики

Перше, що треба зробити — перевірити, що поля відрізняється не більше ніж в одній позиції. Якщо в більше ніж одній, то неможливо за 1 крок змінити два поля, і відповідь `NO`. Якщо два поля рівні, то відповідь `YES`, бо можна отримати поле B не зробивши жодного кроку.

Тепер маємо позицію (x, y) яка відрізняється у двох полях. Перевіримо, чи $A_{x,y} =$ `'.'`, якщо ні, то неможливо поставити знак в непусте поле, і відповідь `NO`. Інакше, подивимось на те, хто мав ходити наступним у полі A, хрестики чи нулики? Це можна зробити порахувавши кількість нуликів і хрестиків, якщо рівно, то хрестики, інакше нулики. Залишається лише перевірити, що поставилось саме те, що треба було.

Задача C. Дискотека

Навчимось перевіряти, чи можна вибрати набір пісень, що в нього відношення суми рейтингу до суми тривалості більше або дорівнює x.

$$\frac{\sum_{i\in X} r_i}{\sum_{i\in X} t_i} \geqslant x$$

$$\sum_{i\in X} r_i \geqslant x \cdot \sum_{i\in X} t_i$$

$$\sum_{i\in X} (r_i - t_i \cdot x) \geqslant 0$$

Таким чином, завжди оптимально брати k пісень, які мають найбільше значення різниці $r_i - t_i \cdot x$.

Зробимо бінарний пошук по відповіді, будемо проходитись по масиву, сортувати й шукати суму перших k. Якщо сума більше ніж 0, то можна отримати відповідь рівну x, інакше ні.

Загальна асимптотика — $\mathcal{O}(N \log A \log N)$.

Задача D. Послідовність

Будемо йти зліва направо по масиву і тримати значення dp_a — максимальна довжина підпослідовності, що закінчується в елементі зі значенням a. Тоді можна перераховувати $dp_{a_i} = \max(dp_{a_i}, dp_j + 1)$, так що $l \leqslant j + a_i \leqslant r$.

Що робити тепер? Перераховуймо dp за допомогою дерева відрізків. Будемо запитувати максимум на відрізку $[l - a_i; r - a_i]$, і оновлювати в точці a_i. Це можна робити за допомогою неявного дерева відрізків, або компресії координат.

Загальна асимптотика — $\mathcal{O}(N \log N)$ з компресією.

Задача E. Матчі

Перефразуємо задачу. Нам дано список ребер, треба розбити на відрізки цей список, щоб враховуючи ребра з кожного відрізку окремо виходив двудольний граф.

Давайте будемо йти й жадібно добирати ребра в граф, поки це ребро нічого не псує. Як ми можемо швидко перевірити, чи псує нам ребро двудольність? Скористаємось СНМ. Будемо в кожній вершинці підтримувати предка і парність довжини шляху до кореня. Таким чином ми "пофарбували"всі вершинки у компоненті.

Коли ми додаємо ребро, що об'єднує дві компоненти, то ми його завжди можемо додати. Інакше, треба перевірити парність відстані до кореня, якщо однакова, то нічого не робимо, інакше розпочинаємо новий відрізок.

СНМ можна модифікувати й перераховувати відстань до кореня під час пошуку кореня. Коли дві компоненти об'єднуються, ми фактично додаємо ребро не $(a; b)$, а $(par[a]; par[b])$. Тому треба сказати, що парність відстані від $par[a]$ до $par[b]$ дорівнює парності суми $dst[a] + dst[b] + 1$.

Автор усіх задач: Андрій Столітній.