Задача A. Ксоня та олімпіада

Автор та розробник: Святослав Бідзіля

Оскільки одна група завжди заходить в 0 секунду, а далі кожні w секунд, поки час не закінчиться. Нескладно помітити, що всього зайде $\lfloor t/w \rfloor + 1$ груп. Тут символами $\lfloor x \rfloor$ позначено округлення вниз — найбільше ціле число менше рівне за x. У кожній групі максимум m людей, а всього людей n (більше за це число людей зайти просто не може), звідки отримуємо пряму формулу для відповіді — $min(n, m \cdot (\lfloor t/w \rfloor + 1))$.

Задача B. Ксоня та алфавітне коло

Автор та розробник: Святослав Бідзіля

Спробуємо обмежити довжину відповіді. Оскільки в англійському алфавіті 26 букв, то відповідь не може бути довшою за 26. Знайдемо для кожного можливого початку довжину найдовшого алфавітного рядка, який з нього починається. Виберемо максимальний. Через обмеження довжини відповіді ми не будемо перевіряти кожний раз більше за 26 символів, отже рішення працює за $26 \cdot n$.

Для спрощення реалізації, можна занумерувати коло від 0 до $n - -- 1$, тоді наступний індекс після індексу i це $(i + 1)\ mod\ n$, де $x \mod y$ — залишок від ділення числа x на число y.

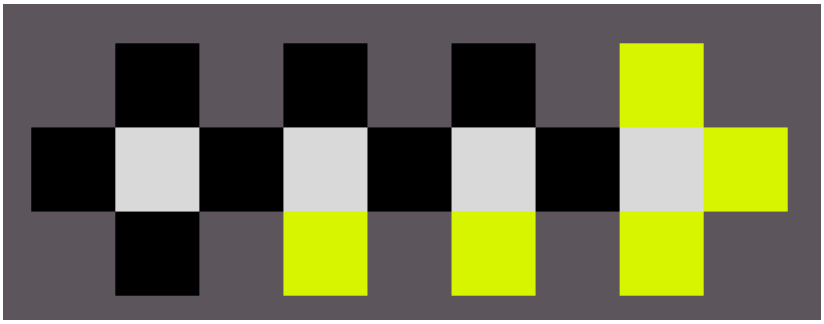
Задача C. Ксоня та двокольорова фігура

Автор та розробник: Святослав Бідзіля

Спочатку знайдемо, коли відповіді немає.

Порахуємо найбільшу площу фігури, в якій x чорних клітинок. Будемо фарбувати нашу фігуру. На першому кроці зафарбуємо будь-яку чорну клітинку в жовтий колір, а всіх її сусідів(білі клітинки) в синій. На кожному наступному колі будемо фарбувати будь-яку чорну клітинку, яка сусідня з синьою в жовтий, і всіх її сусідів в синій. Так, рівно за x кроків ми розфарбуємо всю фігуру (бо чорних клітинок рівно x). На першому колі ми фарбуємо максимум 5 клітинок, на кожному наступному - максимум по 4. Тому, максимальна площа такої фігури - $4x + 1$.

Якщо $4 \cdot min(w, b) + 1 < w + b$ відповіді немає. Інакше, покажемо одну з конструкцій, як можна досягти будь-якої відповіді. Без обмеження загальності, нехай $w < b$. Розташуємо білі клітинки в ряд, через одну. Між ними поставимо чорні клітинки. (Це можливо, бо $w < b$). Далі, всі чорні клітинки, що залишилися можна додати до будь-якої вільної чорної клітинки зверху, знизу, або збоку. Очевидно, що така фігура досягає потрібної площі і подібне рішення нескладно реалізується.



Тут зображений варіант конструкції. Жовтим відмічені місця, куди можна додавати чорні клітинки.

Задача D. Ксоня та граф

Автор та розробник: Святослав Бідзіля

Можна набрати часткові бали розв'язками за допомогою алгоритмів Дейкстри та Флойда, які шукають діаметр в довільному графі.

Щоб отримати вищі бали знайдемо особливість графу з умови. Оскільки граф з $n - 1$ ребром утворює дерево, то граф з n ребер утворює дерево з одним циклом. Будемо розглядати граф, як цикл і кореневі дерева, з коренями в вершинах циклу. Далі є два різних випадки, де може бути діаметр. Або діаметр цього графа знаходиться в одному з дерев, або проходить через цикл.

Для першого випадку нескладно скористатися будь-яким відомим алгоритмом пошуку діаметра в дереві та для кожного дерева знайти діаметр сумарно за $O(n)$.

Для другого випадку важливо зрозуміти, що якщо діаметр проходить через дві вершини циклу (назвемо їх u і v), то та частина діаметра, яка знаходиться в деревах з цими коренями доходить до найглибшого листа в такому дереві. Дійсно, якби вона доходила не до найглибшого листа, можна було б легко покращити відповідь, пішовши від вершини до найглибшого листа.

Знайдемо для кожної вершини циклу глибину найбільшого листа в її дереві (за допомогою dfs-у). Назвемо це число для i-ї вершини h_i. Назвемо відстанню між двома вершинами в циклі $dist(x, y)$ найкоротшу відстань по ребрах циклу між вершинами x і y. Тепер задача звелася до максимізації $h_x + h_y + dist(x, y)$ по усім x, y на циклі.

Як ефективно порахувати $dist(x, y)$? Перенумеруємо вершини циклу від 1 до m. Запишемо всі ваги ребер циклу в масив w, так що вага ребра з вершини i циклу в наступну записана в w_i. Нехай $\sum_i^m w_i = S$. Тоді порахуємо відстань, якби ми йшли по одній стороні циклу $f(i, j) = \sum_{k=i}^{j-1} w_k$. В іншому випадку, ми йдемо по іншій стороні циклу, тому використовуємо всі інші ребра. Тому $dist(i, j) = min(S - f(i, j), f(i, j))$.

Вже зараз задачу можна розв'язати на високий бал за $O(n^2)$, рахуючи шукану величину по всім вершинам циклу.

Для отримання повного балу можна помітити, що при фіксованому i існує таке k, що при всіх вершинах j від i до k $f(i, j) < S - f(i, j)$, а при всіх вершинах j від k до i $f(i, j) >= S - f(i, j)$. Іншими словами, функція монотонна. Для всіх вершин з першої групи відстань дорівнює $f(i, j) = p_{j-1} - p_{i-1}$(Якщо ввести масив p - префіксних сум масиву w), а діаметр на таких вершинах зводиться до $min(h_x + h_y + p_{y-1} - p_{x-1}) = min((h_x - px - 1) + (h_y + p_{y-1}))$. При фіксованому x такий мінімум по всіх j можна легко знайти за допомогою різних структур даних - дерева відрізків, підтримуючи множину усіх кандидатів і т.д. Сума для другої групи вершин схожа і підтримується аналогічно. Для пошуку оптимального k можна використовувати бінарний пошук, або підтримувати k методом двох вказівників.

При використанні сетів або дерева відрізків з'являється додатковий логарифмічний фактор, тому асимптотика рішення $O(n \log n)$.

Задача E. Ксоня та дерево

Автор та розробник: Святослав Бідзіля

Спробуємо розв'язати задачу, якщо в нас вже множина чисел S з піддерева і нам потрібно знайти k-е число з множини всіх можливих XOR-сум підмножин.

У випадку, якщо всі числа є степеню двійки можна помітити, що якщо число є в наборі, то ми можемо або додати його у відповідь, або ні. Так можна зробити з кожним бітом, тому всього буде в множині $2^{|S|}$ чисел. Побудувати k-е число можна жадібно - будемо йти з найбільших чисел до найменших. Якщо ми можемо не ставити якийсь біт (чисел з цим бітом не встановленим $\geqslant k$), то ми не будемо його ставити. Інакше, поставимо цей біт і зменшимо k на кількість чисел, в яких цей біт не встановлений.

Для розв'язання задачі в загальному вигляді потрібно розглянути числа, як набір векторів над полем Z_2. Тоді, xor двох чисел - це додавання цих векторів. Тому, для знаходження всіх можливих xor-ів підмножин, можна побудувати базис векторів, які представляють числа з множини S і подібно до рішення в минулому пункті жадібно встановлювати біти від більшого до меншого. Побудова базису може бути реалізована за $O(\log^3 A)$(Може бути пришвидшено в 32 рази використовуючи бітові операції на числах), а пошук k-го елементу в побудованому базисі - за $O(logA)$.

Тепер, повернемося до оригінальної задачі. Для перших трьох підгруп було достатньо знайти всю множину чисел пошуком вглибину, а далі використати розв'язок з першої частини.

Блок 1 Будь-який повний перебір за $O((n + q)2^n)$ проходив цей блок.

Блок 2-3 Блок проходився якщо перераховувати базиси для всіх вершин за $O(n)$, або знаходити кожний раз заново.

Блок 4 Можна реалізувати задачу для множини чисел, а не дерева.

Блок 5 Якщо в підгрупі немає запитів на зміну числа, то можемо передрахувати всі базиси для кожної вершини дерева - базис вершини це просто базис чисел з базисів її дітей. Це можна порахувати за $O(n \log^2 A)$, а далі відповідати за запити за $O(\log A)$.

Блок 6 Блок вирішувався простішим варіантом базису, достатньо було перевірити наявність потрібного біта в піддереві.

Блок 7 У загальному випадку, можемо звернутися до техніки Ейлерового обходу дерева. Побудуємо за Ейлеровим обходом дерева дерево відрізків, де в кожній вершині буде зберігатися XOR базис її підвідрізка. Для мерджа двох вершин додамо вектори одного базису в інший. Піддерево кожної вершини тепер буде цілісним відрізком на Ейлеровому обході, а отже запит базису на піддереві зводиться до одного запиту до дерева відрізків, а запит зміни числа - до одиничної заміни в дереві відрізків. Реалізація такого дерева відрізків працює за $O(\log^2 A \log n)$ на запит, тому все рішення працює за $O((n+q)\log^2 A \log n)$.