Розбір задачі «Леді та перестановка Козака Вуса»

Давайте підтримувати баланс перестановки. $a_1 = b_1$. Тепер кожного кроку будемо добавляти по два елементи перестановки. Далі, якщо $b_{i-1} = b_i$, то це означає що баланс не змінився, тому дістаємо з сету найменший та найбільший елементи та записуємо в перестановку. Якщо $b_{i-1} < b_i$, то інакше треба добавити b_i, якщо він не добавлений та добавимо найбільший елемент з сету. Якщо $b_{i-1} > b_i$, то треба добавляти найменші елементи з сету.

Розбір задачі «Козак Вус і секрет Леді»

Напишемо динамічне програмування.

dp[i][1] = максимальна відповідь, якщо в позиції і стоїть правіша a[i] з двох, ми отримали бал за пару a[i] і зберегли карту, що стоїть на позиції. Наприклад така послідовність ходів: 1 3 4 2 1(позиція i) 2 -> 1 4 2 1 2 -> 1 2 1 2 -> 2 1 2 -> 2 2

dp[i][0] = максимальна відповідь, якщо ми закінчили в позиції і

Тепер йдемо зліва направо і дивимося: Якщо в позиції, яку зараз перебираємо(i) стоїть лівіша з двох однакових карт то просто dp[i][0] = dp[i - 1][0]

інакше: нехай l[x] - позиція найлівішого входження числа x у масив dp[i][0] = max(dp[l[x] - 1][0] + 1 + o1, dp[l[x] + 1][1] + 1 + o3); dp[i][1] = max(dp[l[x] - 1][0] + 1 + o2, dp[l[x] + 1][1] + 1 + o4); де o1 - кількість однакових карт, які йдуть підряд на проміжку [l[x] + 1, i - 1], o2 - на проміжку [l[x] + 1, i - 2], o3 - [l[x] + 2, i - 1], o4 - [l[x] + 2, i - 2].

Тобто ми можемо або зберегти попереднього або ні; також свою пару, що зліва ми можемо або просто взяти, або взяти як збереженого.

Розбір задачі «XOR шлях»

Примінимо **meet in the middle**. Для кожного шляху з $(1, 1)$ до якоїсь клітинки на головній діагоналі додаємо його ціну(ксор усіх клітинок на шляху) в двійковий **бор** для цієї клітинки. Усього шляхів 2^{n-1}, кожен шлях додається за $O(logA)$ (де A — максимум матриці). Згенеруємо усі шляхи з (n, n) до якоїсь клітинки на головній діагоналі (їх теж буде 2^{n-1}). Коли прийшли в якусь клітинку на головній діагоналі, в ній ми находимо мінімально можливий ксор *зе* значенням нашого шляху з (n, n). Це можно зробити спускаючись з корня по бору цієї клітинки від старших битів до младших і намагаючись робити 0 в поточному розряді. Наприклад якщо в ксорі (з шляху від (n, n)) в поточному розряді стоїть 1, і в нашому борі є перехід по 1, то ми переходимо по 1 (інакше переходимо в 0 і додаємо до поточної відповіді 2 в степені розряду), бо це мінімізує поточну відповідь. Після цього оновлюємо відповідь на задачу *зе* знайденою. Спуск по бору теж працює за $O(logA)$.

Складність $O(2^n \times logA)$.

Розбір задачі «Козак Вус, секрет Леді та відьма»

Напишемо сканлайн зліва направо. Будемо зберігати дві множини горизонтальних відрізків: ті до яких ми можемо дійти, і ті, до яких не можемо. Можуть бути події: починається горизонтальний відрізок(1), закінчується горизонтальний відрізок(2), вертикальний відрізок(3). (1) - просто додаємо в множину недосягнутих відрізків. (2) - видаляємо і дивимось на кого впадемо; якщо на недосягнутий то перекидаємо той недосягнутий у досягнуті (3) - якщо на цій вертикалі є хоча б один досягнутий, то всі робимо досягнуті

Розбір задачі «Автомобільні номери»

Число ділиться на 9, якщо сума всіх його цифр ділиться на 9. Порахуємо суму цифр по модулю 9. Якщо вона менше за 5, то треба відняти цю суму, починаючи зі старшої(першої) цифри. Якщо ж вона більша за 4, то треба додавати цю суму, починаючи з наймолодшої(останньої) цифри.

Розбір задачі «Інвестиції Леді»

Шарнір графу розділяє його на дві окремі компоненти, тому давайте побудуємо block cut tree(далі BCT). Зрозуміло, що Леді не буде інвестувати в компанію, в якій всі вершини не відповідні одній вершині в BCT. Отже, вигідно інвестувати одразу у компанію зі всього набору вершин, відповідних вершині в BCT. Тепер давайте рахувати динаміку на дереві(BCT) dp[v][k][0..1] - максимальна

відповідь для піддерева вершини v, якщо ми інвестували вже в k компаній, та 0 якщо ми не інвестували в компанію вершини v, або 1 якщо інвестували. Останній параметр потрібен, щоб можна було враховувати спільну вершину оригінального графа, яка відповідна двом сусіднім вершинам в ВСТ.

Розбір задачі «Команда»

Давайте для кожної вершини підтримувати сет мінімальних по вазі вершин з її піддерева, сума зарплат яких не перевищує бюджет. Спочатку рішаємо задачу для синів, а далі перераховуємо свій сет, використовуючи метод від меншого к більшому, що дає час роботи $n * logn^2$).

Розбір задачі «Магічний диск і карти»

Напишемо дерево відрізків, у якому будемо зберігати пару(на скільки диск повернеться за годинниковою стрілкою на цьому відрізку(можливо від'ємне число); парність кількості переворотів на цьому відрізку)

Як мерджити два відрізки в один: Нехай (kl; pl) і (kr; pr) - параметри лівого і правого відрізків відповідно, тоді параметри нового відрізка будуть(kl + kr * (1 якщо pl = 0, -1 якщо pl = 1); (pl + pr)

Тепер просто після кожного запиту змінюємо два листка у дереві відрізків і беремо суму на всьому дереві. Далі нескладно знайти відповідь в залежності того додатне чи від'ємне k; p=-1 чи p=1

Розбір задачі «Качка не любить умови»

Давайте зпочатку запам'ятаємо усі позиції нуликів і поставимо в кожну з них значення 1. Зрозуміло, що сума елементів масиву максимальна серед усіх можливих. Тож, якщо сума елементів не є додатньою, то відповідь «`NO`».

Інакше, допоки сума елементів не ділиться націло на k, ми проходимося по позиціях, де колись стояли нулики і ставимо значення елементів на цих позиціях -1 замість 1. Роблячи одну таку "заміну перераховувати суму елементів масиву дуже легко — достатньо відняти 2 від неї.

Якщо ми перебрали усі позиції, де колись були нулі, але так сума елементів масиву не ділиться на k націло, то відповідь «`NO`».

Якщо ми все ж змогли таким чином знайти суму елементів масива, яка ділиться на k, то перевіряємо: якщо ця сума не додатня, то відповідь «`NO`» (зрозуміло, що ми знайшли максимальну можливу суму, що ділиться на k), інакше — виводимо «`YES`» та отриманий масив.

Розбір задачі «Підпослідовності підпослідовностей»

Помітимо, що гарна підпослідовність складається лише з однакових літер. Чому так? Нехай в підпослідовності є дві різні букви. Розглянемо підпослідовність цієї підпослідовності, що складається з цих двох букв. Очевидно, що вона не є паліндромом (неважливо в якому порядку будуть йти ці дві літери).

Як знайти відповідь? Порахуємо кількість кожної літери. Після цього, для кожної букви додамо до відповіді 2 в степені кількості цієї літери мінус один (бо нам підходить будь-яка підпослідовність з цих букв, окрім пустої). Сумарно в нас $|s|$ літер, тому можна 2 в степені рахувати циклом.

Складність $O(|s|)$

Розбір задачі «Цікава»

Якщо змін позицій елементів масиву a немає, то $c_1 \oplus c_2 \oplus \ldots \oplus c_n = a_1 \oplus a_2 \oplus \ldots \oplus a_n$. Помітимо, що зміна позицій елементів з номерами i та j змінить `XOR` масиву c на $a_i \oplus a_j$. Отже відповідь — мінімальне значення $(a_1 \oplus a_2 \oplus \ldots \oplus a_n) \oplus X$, де X — `XOR` парної кількості елементів масиву. Створимо бітовий базис чисел $a_i + 2^{30}$, $1 \leqslant i \leqslant n$. Тоді відповідь можна будувати ітеруючись від більших бітів до менших та перевіряючи, чи можна утворити відповідне число з елементів базису. До чисел додавалось 2^{30} щоб уникнути ситуації коли ми обираємо непарну кількість елементів масиву для створення числа X.

Щоб відповісти на m запитів необхідно знайти базиси на відповідних підвідрізках. Це краще всього зробити за допомогою техніки «`розділяй і володарюй`».

Складність: $O(n \cdot log(n) \cdot log(a) + m \cdot log(a)^2)$ часу та $O(n + m \cdot log(n))$ пам'яті.

Розбір задачі «Козак Вус і нова гра»

Запам'ятаємо для кожного числа позиції, на яких він зустрічається.

Будемо йти зліва направо і зберігати Дерево Відрізків. В кожній вершині зберігаємо відповідь, якщо ми закінчимо в цій позиції. Коли ми проходимо якесь число x, то треба відняти його ціну на проміжку до наступного x не включно. А на проміжку від наступного x до наступного після наступного не включно додати цю ціну.