Оглавление

Три сундука

У Яна есть три сундука с монетами: a, b, c монет, причём $a < b < c$. Можно взять несколько монет из третьего сундука и распределить их между первыми двумя. Можно ли сделать все три сундука равными?

Пусть итоговое одинаковое количество монет в каждом сундуке равно m. Тогда сумма всех монет $a + b + c = 3m$, значит $m = (a + b + c)/3$ должно быть целым.

Из третьего сундука мы можем только забирать монеты, поэтому c может только уменьшаться. Значит, $m \le c$.

В первый и второй сундуки монеты только добавляются, поэтому $m \ge a$ и $m \ge b$. Но так как $a < b < c$, достаточно проверить $m \ge b$.

Итого: условие существования — $a + b + c$ делится на 3 и $m = (a + b + c)/3 \ge b$.

Жадный путешественник

Путешественник идет из левого верхнего угла в правый нижний, двигаясь только вправо или вниз. Он всегда выбирает соседнюю клетку с большей высотой (при равенстве любой). Можно ли заполнить карту числами так, чтобы ни один жадный маршрут не давал максимальную сумму среди всех возможных путей?

Если карта представляет собой одну строку или один столбец ($\min(n, m) = 1$), то существует только один путь — он одновременно и жадный, и максимальный. Такую карту построить нельзя.

Если карта размером 2×2, то есть два пути: (вправо-вниз) и (вниз-вправо). Любой из них можно сделать максимальным, но тогда жадный алгоритм выберет именно его (так как на каждом шаге будет большее число). Сделать так, чтобы максимальный путь не был жадным, невозможно. Значит, для 2×2 ответ тоже NO.

Во всех остальных случаях ($n \ge 3$ и $m \ge 2$ или $m \ge 3$ и $n \ge 2$) можно построить такую карту. Таким образом, условие: ответ YES, если $\min(n, m) > 1$ и $(n, m) \ne (2, 2)$.

Секретное число

Ян задумал число x, приписал к нему справа несколько нулей (хотя бы один) и получил $y = x \cdot 10^k$, где $k \ge 1$. Затем сложил: $n = x + x \cdot 10^k = x(10^k + 1)$. По заданному n найдите все возможные x.

Из условия $n = x(10^k + 1)$ для некоторого $k \ge 1$. Значит, $x = \frac{n}{10^k+1}$ должно быть целым числом.

Перебираем все степени 10^k от 10 до 10^{18} (пока $10^k + 1 \le n$). Для каждого k проверяем, делится ли n на $10^k + 1$. Если да, то добавляем $x = n/(10^k + 1)$ в ответ.

Все найденные x сортируем и выводим.

Минимальная сумма разностей

Дан частично заполненный массив a длины n, где пропуски обозначены -1. Нужно заполнить пропуски неотрицательными целыми числами так, чтобы минимизировать $|\sum_{i=1}^{n-1}(a_{i+1} - a_i)|$. Если несколько способов, выбрать лексикографически наименьший a.

Заметим, что $\sum_{i=1}^{n-1}(a_{i+1} - a_i) = a_n - a_1$. Значит, минимизируем $|a_n - a_1|$. Если первый или последний элемент известен, то второй можно выбрать равным ему, чтобы разность была 0. Если оба известны, то разность фиксирована. Если оба неизвестны, выбираем их равными 0 (это даёт разность 0 и лексикографически минимальный массив). После фиксации a_1 и a_n все остальные элементы заполняем нулями — это не влияет на сумму и даёт лексикографический минимум. В итоге ответом будет $|a_n - a_1|$ и сам массив с заполненными нулями пропусками.

Торговец - легкая версия

За одну сделку можно купить 3^x товаров по цене $3^{x+1} + x \cdot 3^{x-1}$ монет. Нужно ровно n товаров, минимизировать число сделок, а среди таких — стоимость. Найти минимальную стоимость.

Минимальное число сделок достигается, если представлять n в троичной системе счисления и каждую цифру d (0,1,2) обеспечивать соответствующим количеством сделок с данным x. Но так как за одну сделку можно купить ровно 3^x товаров, то для каждой степени x нужно d сделок, где d — цифра в троичном разложении n.

Переводим n в троичную систему, получаем цифры $d_0, d_1, \ldots, d_{k-1}$. Для каждой позиции x (начиная с $x = 0$) стоимость одной сделки равна $3^{x+1} + x \cdot 3^{x-1}$. Умножаем на соответствующую цифру d_x и суммируем. Это и будет минимальная стоимость, так как дробить сделки на меньшие невыгодно (больше сделок), а объединять нельзя (каждая сделка фиксированного размера).

Торговец - сложная версия

За одну сделку можно купить 3^x товаров по цене $3^{x+1} + x \cdot 3^{x-1}$ монет. Нужно ровно n товаров, но можно совершить не более k сделок. Найти минимальную стоимость или -1, если невозможно.

Представим n в троичной системе: $n = \sum d_x \cdot 3^x$, где d_x — цифры 0,1,2. Количество сделок при таком разложении равно сумме цифр $d = \sum d_x$. Если $d > k$, то сразу ответ -1 (не хватает сделок).

Если $d \leq k$, можно уменьшать количество сделок, заменяя одну сделку размера 3^{x+1} на три сделки размера 3^x (это увеличивает число сделок на 2). Наоборот, чтобы уменьшить число сделок, нужно объединять три сделки одного размера в одну большего размера, что уменьшает число сделок на 2.

Пока $d < k - 1$ и есть старшие разряды, переносим единицы из старших разрядов в младшие, увеличивая d на 2 за каждый перенос. После этого либо достигаем $d = k$, либо $d = k - 1$ и после получения нужного представления считаем стоимость как $\sum d_x \cdot (3^{x+1} + x \cdot 3^{x-1})$.

XOR-игра - легкая версия

Даны два массива a и b из 0 и 1. На нечётных ходах Первый может обменять a_i и b_i или пропустить. На чётных — Второй. После n ходов сравниваются XOR-суммы массивов. Определить исход при оптимальной игре.

Заметим, что обмен в позиции i меняет XOR-суммы обоих массивов: a_i и b_i меняются местами. Если $a_i = b_i$, обмен ничего не меняет, поэтому такие позиции не влияют на игру. Интересны только позиции, где $a_i \neq b_i$.

Пусть c — количество таких позиций. Если c чётно, то после всех обменов XOR-суммы будут в итоге равны (доказать можно так: пусть F - ксор сумма первого игрока, а S - второго; если F и S не равны, то их ксор равен 1, а значит, суммарный ксор всех элементов тоже равен 1; но т.к. c - четно, такого быть не могло), а значит, будет ничья.

Если c нечётно, то после всех обменов выиграет тот, на чьём ходу происходит последний обмен.

XOR-игра - сложная версия

Даны два массива a и b длины n. На нечётных ходах Первый может обменять a_i и b_i или пропустить, на чётных — Второй. После всех ходов сравниваются XOR-суммы массивов. Определить исход при оптимальной игре.

Ключевое наблюдение: если $X = \bigoplus_{i=1}^{n} a_i \oplus \bigoplus_{i=1}^{n} b_i \neq 0$, то игроки могут влиять на результат только через позиции, где $a_i \neq b_i$, и только в старшем бите, где X имеет единицу.

Пусть bit — позиция самого старшего бита, в котором X имеет 1. Тогда все позиции, где $a_i \neq b_i$, можно разделить на те, у которых этот бит различается, и те, у которых нет. Игрок, контролирующий последний ход на позиции с различием в этом бите, может обеспечить себе победу.

Если $X = 0$, то после всех обменов XOR-суммы останутся равными (каждый обмен меняет обе суммы одинаково), поэтому ничья.

Отрезки для всех пар

Даны n точек с возрастающими целыми координатами. Для каждой пары (i, j) строится отрезок $[x_i, x_j]$. Для каждого k нужно определить, сколько целых точек принадлежит ровно k отрезкам.

Рассмотрим два типа точек: сами заданные точки и точки между ними. Для точки x_i количество отрезков, которые её покрывают, равно $(i + 1)(n - i) - 1$ (все отрезки, где левый конец $\leq i$, правый $\geq i$, минус вырожденный отрезок $[x_i, x_i]$). Для промежутка между x_i и x_{i+1} длина $d = x_{i+1} - x_i - 1$, каждая из этих точек покрывается $(i+1)(n-i-1)$ отрезками. Подсчитываем, сколько точек имеют каждое значение k, и отвечаем на запросы.

Подсчет точек

На плоскости заданы n кругов с центрами на оси x и радиусами r_i, причём $\sum r_i = m$. Нужно подсчитать количество целочисленных точек (x, y), лежащих внутри или на границе хотя бы одного круга.

Для каждого круга с центром $(x_i, 0)$ и радиусом r_i целочисленные точки (x, y), попадающие в него, удовлетворяют $(x - x_i)^2 + y^2 \leq r_i^2$. Для фиксированного x максимальный $|y|$ равен $\lfloor \sqrt{r_i^2 - (x - x_i)^2} \rfloor$.

Перебираем все возможные x в пределах $[x_i - r_i, x_i + r_i]$ для каждого круга. Для каждого x запоминаем максимальный $|y|$ среди всех кругов, покрывающих этот x. Суммируем по всем x значения $2 \cdot max\_y + 1$.

Так как $\sum r_i = m \leq 2 \cdot 10^5$, суммарное количество перебираемых x по всем кругам не превышает $2m$, что позволяет уложиться в ограничения.

Цветочный мальчик

В саду n цветов с красотой a_i. Нужно собрать m цветов, идя слева направо, так чтобы i-й собранный цветок имел красоту не менее b_i. Можно добавить один цветок с любой красотой k в любое место. Найти минимальное k, позволяющее это сделать, или 0 если можно без добавления, или -1 если невозможно.

Посчитаем, сколько цветов можно собрать без добавления, двигаясь слева направо и жадно беря подходящие под b_j. Получим массив $pre[i]$ — сколько цветов можно собрать из первых i цветов. Аналогично справа налево посчитаем $suf[i]$ — сколько можно собрать из цветов с i по n, но для b с конца.

Если $pre[n] \geq m$, ответ 0.

Иначе перебираем позицию i, куда вставим новый цветок. После вставки слева от него можно взять $pre[i]$ цветов, справа — $suf[i + 1]$ цветов. В сумме $pre[i] + suf[i + 1]$ цветов уже собрано, осталось добрать $x = m - (pre[i] + suf[i + 1])$ цветов. Нам нужно, чтобы $x \leq 1$ (новый цветок может заменить не более одного недостающего). Тогда новый цветок должен быть $b_{pre[i]+1}$ (следующее требуемое значение). Берем минимум таких k по всем i.

Если ни одна позиция не подходит, ответ -1.

Совпадения в массивах

Даны два массива a и b длины n из чисел от 1 до n. Можно удалить одну пару (a_i, b_i) и любое количество раз выполнить операцию: $a_i := b_{i+1}$ или $b_i := a_{i+1}$. Нужно максимизировать количество позиций j, где $a_j = b_j$.

Заметим, что операция позволяет копировать значения из соседней позиции другого массива. Это значит, что если в какой-то позиции i уже есть совпадение или мы можем его сделать, то оно может распространяться влево. Если в позиции n уже есть совпадение ($a_n = b_n$), то можно сделать совпадения во всех позициях, ответ n. Иначе идём справа налево и отслеживаем, можем ли мы обеспечить совпадение в текущей позиции. Условия для совпадения в i:

- $a_i = b_i$ уже есть, или
- $a_i = a_{i+1}$ (тогда можем сделать $b_i = a_{i+1}$), или
- $b_i = b_{i+1}$ (тогда можем сделать $a_i = b_{i+1}$), или
- значение a_i или b_i уже встречалось справа (значит, его можно получить копированием).

Проходим справа налево, помечаем значения, которые мы уже умеем получать. Как только условие нарушается, останавливаемся. Ответ — первый индекс, где ещё можно обеспечить совпадение.

Доставка пиццы

Курьер стартует в (A_x, A_y), должен посетить n точек (x_i, y_i) в любом порядке и вернуться в (B_x, B_y). Двигаться можно только вправо, вверх или вниз (на 1 за шаг). Все x_i строго между A_x и B_x. Найти минимальное время.

Так как двигаться влево нельзя, посещать точки нужно в порядке возрастания x. Группируем точки по x (много точек могут иметь одинаковый x). Для каждого x нужно посетить все точки с этим x, при этом можно двигаться только по вертикали.

Для каждого x находим минимальную и максимальную y среди точек с этим x. Тогда посещение всех точек с этим x требует пройти вертикальный отрезок от y_{min} до y_{max} и обратно, но можно начать с любого конца. Минимальное время для этого x равно $(y_{max} - y_{min}) + \min(|y_{prev} - y_{min}|, |y_{prev} - y_{max}|)$, где y_{prev} — координата, на которой мы пришли с предыдущего x.

Динамика по x: храним два состояния — минимальное время, закончив на y_{min} или на y_{max}. Пересчитываем для следующего x. В конце добавляем путь от последней посещённой точки до (B_x, B_y): нужно пройти по x до B_x (это фиксировано) и по y от текущей позиции до B_y. Итоговое время: $(B_x - A_x)$ + результат динамики.

Максимальное множество

Множество называется красивым, если для любых двух чисел одно делит другое. Для отрезка $[l, r]$ нужно найти максимальный размер такого множества и количество множеств этого размера.

Чтобы множество было красивым, все числа должны лежать на цепочке, где каждое следующее кратно предыдущему. Самый длинный вариант — умножать на 2: $l, 2l, 4l, \ldots$, пока не превысим r. Длина такой цепочки x находится как максимальное x, при котором $l \cdot 2^{x-1} \le r$.

Первый тип множеств: начинаем с некоторого a от l до $\lfloor r/2^{x-1} \rfloor$, затем умножаем на 2 ещё $x - 1$ раз. Количество таких множеств: $\lfloor r/2^{x-1} \rfloor - l + 1$.

Второй тип: заменяем одно из умножений на 2 умножением на 3 (кроме первого шага, чтобы числа оставались целыми). Это возможно, если $l \cdot 2^{x-2} \cdot 3 \le r$. Тогда для каждого из $x-1$ мест, где можно сделать замену, считаем количество подходящих начальных a: $\lfloor r/(2^{x-2} \cdot 3) \rfloor - l + 1$. Добавляем это к ответу.

Если $l \cdot 3 > r$, то второй тип невозможен, выводим только первый.

Граф и граф

Даны два связных графа на n вершинах. Фишки стартуют в s_1 и s_2. На каждом шаге фишки перемещаются в соседние вершины (каждая в своём графе). Стоимость шага $|u_1 - u_2|$. Нужно найти минимальную суммарную стоимость за бесконечное время или -1, если она бесконечно велика.

Состояние игры — пара (v_1, v_2) — текущие положения фишек. Переходы: из (v_1, v_2) можно перейти в (u_1, u_2), где u_1 сосед v_1 в первом графе, u_2 сосед v_2 во втором, с весом $|u_1 - u_2|$.

Если существует цикл отрицательного суммарного веса в этом графе состояний, то стоимость можно бесконечно уменьшать, ответ $-\infty$ (в задаче -1). Иначе минимальная стоимость достигается на каком-то конечном пути.

Запускаем алгоритм Дейкстры на графе состояний (вершин n^2). Старт — (s_1, s_2). Если найдём состояние, где $v_1 = v_2$, то можно остановиться — дальше двигаться не обязательно. Ответ — минимальное расстояние до такого состояния.

Если очередь опустела, а такое состояние не достигнуто, то ответ -1 (бесконечная стоимость, так как нельзя достичь равенства).

Необычный город

Дано дерево из n вершин, рёбра чёрные или красные. Последовательность вершин длины k называется хорошей, если кратчайший путь между соседними вершинами в последовательности проходит хотя бы по одному чёрному ребру. Найти количество хороших последовательностей.

Общее количество последовательностей длины k из n вершин равно n^k. Вычтем плохие — те, где все пути между соседями проходят только по красным рёбрам.

Удалим все чёрные рёбра из дерева. Останутся компоненты связности из красных рёбер (возможно, изолированные вершины). Внутри каждой такой компоненты любые два пути между вершинами состоят только из красных рёбер. Значит, если все вершины последовательности лежат в одной такой компоненте, то последовательность плохая.

Для каждой компоненты размера s количество плохих последовательностей, целиком лежащих в ней, равно s^k. Суммируем по всем компонентам. Тогда ответ: $n^k - \sum s^k$ по модулю $10^9 + 7$.