Разбор задач

Ваня и сложный шифр (7-8 классы)

Данная задача решается полным перебором возможных цифр и их кодов.

Начиная от начала заданного зашифрованного числа, будем двигаться по нему слева направо и перебирать коды. При первом же различии символов текущий код будем отбрасывать и пытаться совместить следующий. На одном из 10 возможных попыток мы найдем цифру, причем единственную, код которой будет соответствовать текущему фрагменту строки. Необходимо на этом шаге вывести найденную цифру и начать поиск следующей цифры с позиции, следующей за расшифрованным на текущий момент фрагментом строки.

Из определения префиксного кода следует, что число будет расшифровано однозначно.

Иннокентий и химия (7-11 классы)

Первая подзадача содержит тест из условия. Ее необходимо решить, для того чтобы засчитались баллы за остальные подзадачи.

Подзадача 2: $4 \leqslant N \leqslant 500$, $N - 2 \leqslant M \leqslant N$

Для восстановления ответа сохраним порядок координат отверстий, который представлен во входных данных.

В `C++` для этого координаты можно хранить в `std::pair`, где ключ — это порядковый номер координаты во входных данных, значение — координата отверстия.

Отсортируем координаты по возрастанию. Создадим массив из $n - m$ нулей и m единиц, моделирующий множество заполненных отсеков. Например, если i-ый индекс массива равен нулю, то в данный отсек не будет перелит реактив, иначе этот отсек будет содержать какой-то из реактивов. Тогда можно перебрать все возможные расстановки элементов данного массива и найти, какое из минимальных расстояний между отверстиями будет наибольшим.

Подзадача 3: $2 \leqslant N \leqslant 1000$, *координаты не превосходят* 10^4

Отсортируем координаты по возрастанию. Будем искать ответ линейно. Для этого выберем отверстие с наибольшей координатой X и будем итерироваться от X до 1 с шагом -1. Для проверки допустимости расстояния попробуем перелить реактивы в отсеки так, чтобы между всеми было расстояние, как минимум i. Данную проверку можно выполнить жадно за линейное время. Перельем первый реактив в самый первый сосуд, далее остальные будем переливать в сосуды с шагом, как минимум равным i. Первое подходящее число и будет наибольшим минимальным расстоянием.

Подзадача 4: *нет дополнительных ограничений.*

Улучшим решение подзадачи 3. Теперь вместо линейного поиска будем использовать бинарный поиск по ответу с левой границей 1 и правой границей X.

Болотце (7-11 классы)

Первая подзадача содержит тест из условия. Ее необходимо решить, для того чтобы засчитались баллы за остальные подзадачи.

Подзадача 2: *На болоте нет препятствий*

Простой способ понять, в какой точке мы находимся — выбрать один из углов плана болотца и направиться к нему. Например, можно идти по плану влево, пока не получим ответ интерактора `Impossible`, после чего повторить процедуру, направившись вверх. Таким образом, мы точно узнаем, что находимся в левом верхнем углу болота.

Подзадача 3: *Встречаются только кочки*

Чтобы решить эту и дальнейшие подзадачи, отметим важный факт из условия: препятствия **никак не касаются** ни друг друга, ни границ болота. Как следствие — Жабка может обойти его с любой стороны.

Заметим, что любой угол задает нам пару направлений (например, `(L, U)` для левого верхнего угла), которых следует придерживаться при перемещении. Будем использовать следующую идею: продолжать идти в каком-то направлении, пока это возможно, а затем менять направление на второе из заданной пары. Для решения данной подзадачи этого достаточно. Необходимо лишь проверить, что при смене двух направлений подряд Жабка не сделала ни одного шага — это означает, что Жабка достигла угла.

Подзадача 4: *Встречаются только кувшинки*

Для решения подзадачи с кувшинками решение предыдущей подзадачи необходимо доработать, так как Жабка может попасть в угол кувшинки. Необходимо добавить проверку на это условие, а именно, сделать шаг назад и сменить направление.

Подзадача 5: *Могут встретиться оба вида препятствий*

Стоит отметить, что полное решение будет получать вердикт `WA` на тестах с маленьким размером болота, т.к. ему не будет хватать количества запросов. Не трудно заметить, что кувшинка может встретиться только на поле, у которого обе размерности больше или равны 5, поэтому требуется добавить в решение отсечение лишних проверок.

Треугольники (7-11 классы)

Первая подзадача содержит тест из условия. Ее необходимо решить, для того чтобы засчитались баллы за остальные подзадачи.

Подзадача 2: $1 \leqslant N \leqslant 100$

Для треугольника, построенного на каждой тройке точек, проверяем наличие пары равных сторон, наличие стороны длины t и равенство площади S. Площадь можно вычислить через векторное произведение. Сложность решения $O(N^3)$.

Подзадача 3: $1 \leqslant N \leqslant 5000$

Сторона длины t в равнобедренном треугольнике может находиться либо в основании, либо быть боковой стороной. При этом одновременно и основание, и боковая сторона такую длину иметь не могут, так как в таком случае треугольник будет равносторонним, а равносторонних треугольников, у которых все вершины имеют целочисленные координаты, не существует. Таким образом, можно посчитать количество треугольников этих двух видов (где t — длина основания и где t — длина боковой стороны) отдельно и сложить результаты для получения ответа.

- Для подсчета числа треугольников, у которых t — длина основания, можно заметить, что из равнобедренности искомого треугольника третья вершина должна лежать на серединном перпендикуляре к основанию, что с учётом ограничения на площадь даёт два варианта расположения третьей вершины, и остается только проверить, существует ли такая точка (например, используя std::unordered_set). Асимптотика этой части $O(N^2)$.

- Для подсчёта числа треугольников, у которых t — длина боковой стороны, выпишем для каждой точки список всех точек, находящихся на расстоянии t от неё, переберем все такие пары точек и проверим площадь каждого полученного треугольника. Можно

доказать, что суммарная длина полученных списков $O(N)$, соответственно перебор пар займёт $O(N^2)$ операций.

Таким образом, общая сложность решения $O(N^2)$

Подзадача 4: $1 \leqslant N \leqslant 10^5$

Заметим, что при ограничениях координат до 10^5 для каждой точки с целочисленными координатами существует не более $C = 400$ кандидатов точек с целочисленными координатами таких, что расстояние между ними равно некоторому заданному значению t. При этом, множество кандидатов для одной точки можно получить из множества кандидатов для другой точки смещением: фактически каждый кандидат определяется парой $(\Delta x, \Delta y)$ такой, что $\Delta x^2 + \Delta y^2 = t^2$.

Можно предварительно вычислить всех кандидатов (пары $(\Delta x, \Delta y)$) за $O(t)$ операций. Далее, используя их, можно сократить сложность подсчета числа треугольников, у которых t — длина основания, до $O(NC)$ следующим образом: перебрать для каждой точки всех кандидатов, для каждого проверить существование точки с этими координатами, после чего восстановить координаты третьей точки и проверить её существование.

Также можно предварительно перебрать пары кандидатов за $O(C^2)$ операций и выписать только те, при взятии которых, как сторон равнобедренного треугольника, площадь полученного треугольника будет равна S. Нетрудно заметить, что таких пар будет $O(C)$: из площади и длин боковых сторон можно однозначно вычислить синус угла между ними. Поэтому, зафиксировав одного кандидата в паре, мы можем выбрать второго кандидата не более чем двумя способами. Перебирая для каждой точки только такие пары, получаем ответ за $O(NC)$

Итоговая сложность решения $O(C^2 + NC)$

Задачи на геометрию (7-11 классы)

Первая подзадача содержит тест из условия. Ее необходимо решить, для того чтобы засчитались баллы за остальные подзадачи.

Подзадача 2: $0 < N \leqslant 10^{18},\ 0 \leqslant M = K \leqslant 200$

Заметим, что в любом соревновании может быть любое количество задач. Итоговое количество различных последовательностей соревнований равно $(M+1)^N$. Это значение можно посчитать с помощью быстрого возведения в степень за $O(logN)$ операций.

Подзадача 3: $N \cdot M \cdot K \leqslant 5 \cdot 10^7$

Для решения данной подзадачи понадобится динамическое программирование.

Обозначим через $dp[i][j]$ — количество последовательностей длины i, таких что в последнем соревновании ровно j задач.

Тогда значение $dp[i+1][j]$ можно посчитать следующим образом:

$dp[i+1][j] = \sum_{j'=max(0,j-k)}^{min(M,j+k)} dp[i][j']$.

Это решение работает за $O(N \cdot M \cdot K)$ операций.

Подзадача 4: $N \cdot M \leqslant 5 \cdot 10^7$

Значения $dp[i][j]$ можно пересчитывать следующим образом:

$dp[i][j] = dp[i][j-1] - dp[i-1][j-k-1] + dp[i-1][j+k]$.

Пересчитывать значение динамики можно с помощью скользящего окна, уменьшив время работы до $O(N \cdot M)$.

Также, из-за ограничений по памяти, стоит хранить значения только для последних двух «слоев» динамики.

Подзадача 5: $0 < N \leqslant 10^{18},\ 0 \leqslant M \leqslant 200,\ 0 \leqslant K \leqslant M$

Полностью задачу можно решить с помощью возведения матрицы в степень.

Пусть V_i — вектор-столбец значений i-го «слоя» динамики, а A — матрица перехода между слоями: $V_{i+1} = A \cdot V_i$.

Тогда последний слой можно получить следующим образом: $V_N = A^N \cdot V_0$.

Здесь V_0 — единичный вектор.

Матрица A задается следующим образом: $A_{ij} = 1 \longleftrightarrow |i - j| \leqslant K$.

Умножение матриц работает за $O(M^3)$ операций, а итоговое решение с быстрым возведением матриц в степень работает за $O(M^3 log(N))$.

Заоблачная экономия (9-11 классы)

Первая подзадача содержит тест из условия. Ее необходимо решить для того, чтобы засчитались баллы за остальные подзадачи.

Подзадача 2: $1 \leqslant N, M \leqslant 7$

В этой подзадаче достаточно было написать полный перебор, попробовать разместить каждый файл в каждом из сервисов.

Сложность такого решения $O(m^n)$.

Подзадача 3: *все* $r_j = 0$

Здесь можно было написать жадное решение. Для каждого файла выбираем сервис с достаточным максимальным размером и минимальной ценой размещения одного файла. Для этого можно было, например, отсортировать файлы и сервисы по убыванию размеров и пройти по ним двумя указателями, поддерживая сервис с минимальной ценой.

Сложность такого решения $O(m \log m + n \log n)$.

Подзадача 4: $1 \leqslant N, M \leqslant 500$

Привяжем каждый файл к сервису с ближайшим сверху максимальным размером файла. Теперь отсортируем сервисы по убыванию максимального размера. Если мы регистрируемся в каком-то из сервисов, то им же мы можем покрыть файлы, которые привязаны к последующим. Задача сводится к поиску оптимального разбиения сервисов на подотрезки.

Пусть $cnt(x, y)$ — количество файлов, привязанных к сервисам, лежащим в промежутке между сервисом x и y включительно. Тогда цена покрытия отрезка $[x, y]$ сервисом x равна

$$f(x, y) = \begin{cases} r_x + p_x \cdot cnt(x, y), \text{ если } cnt(x, y) > 0, \\ 0, \text{ иначе} \end{cases}$$

Пусть $dp[i][k]$ — цена разбиения отрезка $[0, i]$ на $k + 1$ подотрезков.

Тогда $dp[i][0] = f(0, i)$, $dp[i][k] = \min\limits_{0 \leqslant j < i} dp[j][k - 1] + f(j + 1, i)$.

Ответ на задачу — $\min\limits_{0 \leqslant j < m} dp[m - 1][j]$.

Обратным ходом найдём нужное распределение.

Сложность такого решения $O(m^3 + n \log n)$.

Также можно было написать другие виды динамик, например, за $O(n^2 m)$.

Подзадача 5: $1 \leqslant N, M \leqslant 5000$

$dp[i] = f(0, i)$, $dp[i] = \min\limits_{0 \leqslant j < i} dp[j] + f(j + 1, i)$.

Подзадача 6: $1 \leqslant N, M \leqslant 10^5$

Применим метод convex-hull-trick (CHT) к решению из предыдущей подзадачи.

Очевидно, что если к каждому из просматриваемых значений мы прибавим константу, то индекс для минимума не поменяется. Сам минимум, конечно, увеличится на эту константу. Нашей константой будет $cnt(i+1, m-1) \cdot p_i$.

Получится: $dp[i] = \min\limits_{0 \leqslant j < i} dp[j] + f(j+1, i) + cnt(i+1, m-1) \cdot p_i = \min\limits_{0 \leqslant j < i} dp[j] + f(j+1, m-1)$.

Для использования СНТ $dp[i]$ представляет прямую, проходящую через точки $(0,\ dp[i])$ и $(1,\ dp[i] + N - cnt(j+1, m-1))$.

Робот-строитель (9-11 классы)

Первая подзадача содержит тест из условия. Ее необходимо решить, для того чтобы засчитались баллы за остальные подзадачи.

Подзадача 2: $1 \leqslant N, M \leqslant 3$

Для решения второй подзадачи достаточно перебрать всевозможные покраски кубиков и посмотреть, сколько из них являются симметричными после перестановки кубиков. Это можно сделать за $O(M^N \cdot N!)$.

Для решения дальнейших подзадач введем следующие обозначения: класс кубиков — это множество кубиков, которые после какого-то (возможно, нулевого) количества перестановок роботом должны иметь одинаковый цвет. Тогда ответом всегда будет M^C, где C — количество классов эквивалентности.

Подзадача 3: $1 \leqslant N, M \leqslant 10$

Для этой подзадачи можно явно разбить на классы все кубики с помощью применения перестановки и явного хранения классов кубиков.

Итоговое время работы: $O(N! \cdot N^2)$

Подзадача 4: $1 \leqslant N, M \leqslant 100$

В данной подзадаче можно использовать систему непересекающихся множеств (СНМ) без дополнительных оптимизаций. Итоговое время работы $O(N! \cdot N)$.

Далее будем строить решения для прохождения оставшихся групп тестов.

Представим операцию перестановки в виде графа: вершина u соединена с вершиной v если и только если $p_u = v$. Введем также «сдвиг» каждой из вершин циклов относительно некоторой начальной вершины цикла (соответствует количеству применений перестановки, чтобы попасть в соответствующую позицию). Далее дополним граф следующим образом: соединим циклы с учетом сдвигов, если они стоят на симметричных позициях.

Можно заметить следующие факты:

1. Количество классов в цикле делит его длину нацело.

2. У соединенных циклов одинаковое количество классов.

3. Количество классов делит нацело модули разностей сдвигов соединенных вершин.

Подзадача 5: $1 \leqslant N, M \leqslant 1000$

Для решения пятой подзадачи можно изначально ограничить количество классов, как НОД$(len(a), len(b), \ldots)$,

где НОД — наибольший общий делитель; $len(x)$ — длина цикла x; а a, b, $\ldots$ — соединенные циклы.

После этого, явно присвоим классы каждому элементу цикла, после чего приравняем те классы, кубики которых бывают на симметричных позициях.

Итоговое время работы $O(N^2 log(N))$, поскольку в худшем случае будет один цикл, а значит, количество классов в нем — N, следовательно, для объединения может потребоваться не более $N-1$ применения перестановки, и $O(log(N))$ времени потребуется для системы непересекающихся множеств.

Подзадача 6: $1 \leqslant N, M \leqslant 2 \cdot 10^5$*;* $p_i = N + 1 - i$

Можно заметить, что меняются местами всегда те кубики, цвета которых должны совпадать. Тогда ответ — $O(M^{N/2})$.

Подзадача 7: *Нет дополнительных ограничений*

Для полного решения будем в системе непересекающихся множеств хранить не сами классы кубиков, а объединения циклов. Выделим циклы, добавим «сдвиги» вершинам и переберем вершины с номерами $1, 2, \ldots N/2$. Пусть сейчас выбрана вершина i.

Рассмотрим два случая: циклы, которым принадлежат вершины с номерами i и $N-i$, находятся в разных множествах или в одном.

В первом случае просто объединим множества этих циклов, обновив количество классов для полученного множества равным НОД(a, b), где a и b — количество классов для исходных множеств. Также установим значение сдвига присоединяемого множества как сдвиг множества, которое присоединяют, минус сдвиг вершины, которую присоединяют.

Во втором случае необходимо обновить количество классов внутри множества на разницу сдвигов вершин.В конце сумма количеств классов по всем множествам будет равна итоговому количеству классов кубиков.

Итоговое время работы: $O(N \cdot log(N))$, поскольку будет всего лишь $N/2$ операций объединения множеств, и каждое из них будет обрабатываться за $O(log(N))$ при использовании СНМ.

Также для решения первых шести подзадач можно использовать явную симуляцию перестановки кубиков и поддержки классов кубиков с помощью системы непересекающихся множеств с оптимизацией сжатия путей.

Это работает за $O(N \cdot log(N) \cdot C)$, где C — количество необходимых перестановок.