

Разбор задач 7-8 классы

Ваня и сложный шифр

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

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

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

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: $2 \leq N \leq 1000$, 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

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: $1 \leq N \leq 100$

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: $1 \leq N \leq 5000$

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: $1 \leq N \leq 10^5$

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

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

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

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 $O(\log N)$ операций.

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

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: $N \cdot M \leq 5 \cdot 10^7$

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: $0 < N \leq 10^{18}$, $0 \leq M \leq 200$, $0 \leq K \leq M$

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: $A_{ij} = 1 \iff |i - j| \leq K$.

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