

Олимпиада имени И.М. Дризе по программированию

Разбор задач

Задача А. Самое короткое условие

Авторы задачи: Мосолов Владислав

Подзадача 1

По условию, на отрезке от 1 до R находится ровно одно число, делящееся на y . Если $x \neq 1$, то ответ равен 0.

Очевидно, что минимальное число, которое делится на y , и есть сам y . Тогда нам подходят все отрезки, для которых выполняются условия:

$$1 \leq l \leq y \quad y \leq r \leq R$$

.

Количество способов выбрать такие отрезки в точности равно: $y \cdot (R - y + 1)$.

Подзадача 2

Числа, которые делаются на y имеют вид: $k \cdot y$, где k — натуральное. Тогда количество чисел, делящихся на y на отрезке от 1 до d равно: $\text{cnt}(d) = \left\lfloor \frac{d}{y} \right\rfloor$. Таким образом, количество чисел, делящихся на отрезке $[l, r]$ на y равно:

$$\text{cnt}(r) - \text{cnt}(l - 1) = \left\lfloor \frac{r}{y} \right\rfloor - \left\lfloor \frac{l - 1}{y} \right\rfloor$$

Для решения данной подзадачи можно перебрать всевозможные отрезки и посчитать количество таковых, где $\text{cnt}(r) - \text{cnt}(l - 1) = x$.

Асимптотика: $O(R^2)$.

Подзадача 3

Для решения данной подзадачи будем перебирать возможные начала отрезков l и для каждого начала будем считать количество подходящих концов r , что $l \leq r \leq R$.

Найдем первое подходящее r_1 , что на отрезке $[l, r]$ в точности x чисел, кратных y . Нетрудно заметить, что $y \mid r$. Также найдем такое r_2 , что на отрезке $[l, r]$ в точности $x + 1$ число, кратное y . Тогда наше подходящее r заключено между r_1 и $r_2 - 1$, значит, всего подходящих концов будет в точности $\min(R, r_2) - \min(R, r_1)$.

Заметим, что n -ное число, кратное y , равно $n \cdot y$, поэтому $r_1 = y \cdot (\text{cnt}(l - 1) + x)$, а $r_2 = r_1 + y$.

Асимптотика: $O(R)$.

Подзадача 4

Заметим, что если левая граница лежит в промежутке $[ky + 1, (k + 1)y]$, то правая граница должна лежать в промежутке $[(k + x)y, (k + x + 1)y - 1]$, чтобы на отрезке было в точности x чисел, кратных y . Данные диапазоны будем называть левым и правым соответственно.

В таком случае, каждому левому диапазону можно сопоставить ровно один правый диапазон. Длины всех левых и правых диапазонов равны y , кроме самого правого диапазона, который может не полностью в границы $[1, R]$, его длина будет равна: $R \% y + 1$.

Всего левых диапазонов: $\text{cnt}(R) - x + 1$. Из них $\text{cnt}(R) - x$ имеют длину y , последний - $R \% y + 1$. В таком случае, получаем итоговую формулу:

$$\left(\left\lfloor \frac{R}{y} \right\rfloor - x \right) \cdot y^2 + y \cdot (R \% y + 1)$$

Задача В. Найди отрицательное

Авторы задачи: Вахранев Андрей, Мосолов Владислав

Подзадачи 1-3

Подзадачи решаются просто разбором случаев. Для всех этих подзадач можно обобщить одно решение: рассмотрим самую верхнюю и самую нижнюю точки среди $(x_1, y_1 + r_1), (x_1, y_1 - r_1), (x_2, y_2 + r_2), (x_2, y_2 - r_2)$. Если обе точки принадлежат обеим окружностям, тогда ответ -1 , поскольку окружности совпадают. Иначе хотя бы 1 из этих точек лежит вне второй окружности, ее и выведем (случай, когда обе точки лежат внутри обеих, означает факт совпадения окружностей).

Подзадача 4

Подойдет любая точка внутри одной из окружностей. Можно вывести центр.

Подзадача 5

Можно перебрать $O(C \cdot C) = O(C^2)$ всевозможных точек и проверить каждую из них, где $|C| \leq 10^3$

Подзадача 6

Данная подзадача решается более умным способом, где мы перебираем только одну из двух координат. Авторы задачи не очень хотят детально расписывать, какие вторые координаты в таком случае нас интересуют на рассмотрении, поэтому упражнение остается читателю. Достаточно поверить, что их константное число.

Асимптотика: $O(C)$, где $C \leq 10^6$

Подзадача 7

Без ограничения общности, скажем, что $r_1 \geq r_2$. Пусть окружности не совпадают (если это так, выведем -1). Рассмотрим две точки, принадлежащие первой окружности: $A = (x_1 + r_1, y_1), B = (x_1 - r_1, y_1)$. Докажем, что обе точки не могут одновременно принадлежать второй окружности.

Предположим, что точки A и B все же принадлежат второй окружности. Расстояние между точками равно $2r_1$. Значит, диаметр второй окружности не меньше, чем $2r_1$, то есть $r_2 \geq r_1$. Получаем, что $r_1 = r_2$, при этом точки A и B принадлежат обеим окружностям, из чего делается вывод о том, что окружности совпадают, чего не могло быть. Значит, достаточно рассмотреть лишь две точки.

Задача С. Футбол в Берляндии

Авторы задачи: Вахранев Андрей

Подзадачи 1-2

В данных подзадачах достаточно перебрать всевозможные варианты раздать майки и для каждого из вариантов проверить, удовлетворяет ли он условию задачи. Конечно же, во второй подзадаче нужен предподсчет.

Асимптотика: $O(n! \cdot n)$

Подзадачи 3-4

Для решения данных подзадач можно использовать метод динамического программирования

База: одному игроку есть только один способ выдать майку, поэтому $dp[1] = 1$

Переход: предположим, что мы уже подсчитали ответы для всех чисел от 1 до $n - 1$. Теперь посчитаем ответ для n .

Рассмотрим, какому игроку можно сопоставить майку с номером $n + 1$. Подходит любой такой номер игрока x , что $x \leq n$ и $x \mid (n + 1)$.

Майки с номерами от 2 до x мы можем сопоставить только игрокам с номерами от 1 до $x - 1$, поскольку номер новой майки не может быть меньше номера старой майки, а игрок с номером x уже занят. Тогда майки с номерами от $x + 1$ до n достанутся игрокам с номерами от $x + 1$ до n , притом легко заметить, что у каждого игрока в данном интервале номер майки совпадет со старым номером майки (если мы будем раздавать майки, начиная с конца, то это интуитивно понятно)

Получается, что количество способов распределить n маек между игроками, если майка с номером $n + 1$ достанется игроку x , равно количеству способов распределить майки с номерами от 2 до x среди игроков с номерами от 1 до $x - 1$, что мы уже насчитали до этого. Итоговая формула:

$$dp[n] = \sum_{i \leq n \text{ \& } i \mid (n+1)} dp[i]$$

Где i мы перебираем от 1 до n . В подзадаче 4 мы просто делаем предподсчет.

Асимптотика: $O(n^2)$

Подзадачи 5-6

Делаем все то же самое, что и в подзадачах 4-5, только перебор делителей числа $n + 1$ делаем более умно, за $O(\sqrt{n})$. В подзадаче 6 нужен предподсчет.

Асимптотика: $O(n\sqrt{n})$

Задача D. Посчитай-ка пути

Авторы задачи: Вахранев Андрей, Мосолов Владислав

Подзадача 1

Для каждого запроса вида $\{u, v, d\}$ запустим алгоритм **bfs** от вершины u , параллельно которому храним массив $prev$ и sum , где $prev[i]$ означает вершину, из которой мы перешли в i , а $sum[j]$ равно сумме весов на ребрах на пути до j .

Легко восстанавливается путь $u \rightarrow v$. Он выглядит как: $v \leftarrow prev[v] \leftarrow prev[prev[v]] \leftarrow \dots \leftarrow u$. Сохраним данный путь и будем идти по нему по порядку, проверяя на каждой вершине условие $sum[i] \cdot (sum[v] - sum[i]) = d$, где i - это всевозможные вершины на пути (кроме v), которые будут делить наш путь. Тогда для каждого запроса просто выведем количество подходящих вершин.

Асимптотика: $O(nq)$

Подзадача 2

Важный факт в данной подзадаче: расстояние между любой парой вершин в данном дереве не превосходит $2 \cdot \log n$

На самом деле, подзадача решается таким же способом, как и подзадача 1, отличие лишь в том, что путь $u \rightarrow v$ нужно искать более эффективно, не через `bfs` за $O(n)$. Как это сделать? Давайте просто, как при насчитывании бинарных подъемов в дереве, для каждой вершинки сохраним `tin` и `tout`, время входа и выхода, а затем будем подниматься по родителям вершины u до тех пор, пока не окажется, что вершинка v не лежит в поддереве этого родителя anc , что как раз можно проверить через условие: $tin[anc] \leq tin[v] \leq tout[v] \leq tout[anc]$ (на самом деле, данная вершина anc будет не что иное, как наименьший общий предок вершин u и v). И далее, наш путь от u до v просто будет подниматься до anc , а потом спускаться.

Асимптотика: $O(n + q \log n)$

Подзадача 3

В данной подзадаче мы уже должны реализовать бинарные подъемы и функцию поиска `lca`.

Выпишем веса на нашем пути от u до v : $w_1, \dots, w_k$. Разобьем наш путь на два в некоторой вершинке, получим разбиение на два множества весов: $w_1, \dots, w_i$ и $w_{i+1}, \dots, w_k$. Пусть сумма на первом пути a , на втором - b , то есть $a = \sum_{j=1}^i w_j$, $b = \sum_{j=i+1}^k w_j$. В данной подзадаче мы знаем, что $w_i = 1 \forall i$, то есть $a + b = dist(u, v)$, где $dist(u, v)$ - расстояние от u до v по ребрам. Итого, мы получаем систему:

$$\begin{cases} a + b = dist(u, v) \\ a \cdot b = d \end{cases}$$

Данная система сводится к квадратному уравнению, у которого мы просто ищем решения, затем выводим количество решений. Случай, где $i = 0$, то есть случай с одним путем, нужно рассмотреть отдельно.

Асимптотика: $O((n + q) \log n)$

Подзадача 4

Здесь нужны наблюдения из подзадачи 3. Поскольку теперь веса произвольные, введем обозначение $sum(u, v)$ - сумма весов на пути от u до v . Тогда наша система превращается в:

$$\begin{cases} a + b = sum(u, v) \\ a \cdot b = d \end{cases}$$

Аналогично, найдем все решения (рассмотрев отдельно случай с $i = 0$). Допустим, нашли решение (x, y) . Что логически оно значит? Нам нужно посчитать количество способов взять некоторый префикс пути от u до v так, чтобы сумма весов на нем была равна x . Другой смысл: посчитать количество суффиксов так, чтобы сумма весов на нем до v была равна y .

Найдем $lca(u, v)$ (наименьший общий предок). Заметим, что вершина, которая делит наш путь на два, находится либо на участке (u, lca) , либо (lca, v) , либо совпадает с lca , где за (c, d) мы обозначаем невключаемый с обеих сторон путь от c до d . Тогда наша задача разбивается на следующие: посчитать количество способов взять некоторой префикс пути от u до lca (здесь мы будем включать lca для удобства) такой, что сумма на нем равна a , затем посчитать количество способов взять некоторой суффикс пути от v до lca , на котором сумма равна b , и, наконец, вычесть 1 в том случае, если разбиение пути строго в lca нам подходит (так как мы его учли два раза).

Давайте считаем все наши запросы и будем их обрабатывать по ходу `dfs`. Будем хранить `map<long long, vector<int>> w`. Каждый раз, входя в новую вершину i , будем добавлять по ключу `sum` (храним сумму на пути от корня до вершины i) в нашу структуру вершину i , то есть `w[sum].push_back(i)` (аналогично, когда обойдем все поддерево, будем делать `pop_back`). Наша структура в каждый момент времени будет отвечать за всевозможные суммы на пути от корня до всех вершин, которые есть среди предков вершины i (притом внутри структуры вершины хранятся по увеличению глубины в обходе).

Пусть теперь мы находимся в вершине i , обрабатываем там запросы. Пришел запрос вида (lca, a) , который означает, что мы хотим посчитать количество вертикальных путей от i до некоторого предка (не выше lca) таких, что сумма на них равна a . Задача эквивалентна задаче посчитать количество вершин в векторе $k[sum - a]$ (где `sum`, опять же, сумма на пути до корня), что эти вершины лежат не выше lca . Поскольку вершины упорядочены по глубине, задача легко решается при помощи бинарного поиска.

Асимптотика: $O(n \log n + q \log^2 n)$ или $O((n + q) \log n)$, если вместо использования `map` мы изначально сожмем всевозможные суммы на путях до корня, а затем мы будем использовать массив векторов.

Задача Е. Гробовая геометрия

Авторы задачи: Вахранев Андрей, Мосолов Владислав

Подзадача 1

Для удобства добавим ко всем координатам 10, чтобы они стали неотрицательными. Далее, раз мы знаем, что существует такой ответ (a, b) , что $-10 \leq a, b \leq 10$, будем перебирать всевозможные точки на плоскости и проверять для них, выполняется ли условие с квадратами расстояний. Обозначим за $C \leq 20$ максимально возможное значение по координате среди всех точек. В таком случае, перебор можно написать за $O(t \cdot C^4)$, где t - разбор всех тестов, C^4 появляется из комбинации перебора возможной точки за C^2 и проверки выполнения условий за C^2 (где мы пробегаем всю решетку, смотрим, сколько в каждой точке точек из входных данных и аккуратно пересчитываем)

Асимптотика: $O(t \cdot C^4)$

Подзадача 2

Рассмотрим две первые вершины, обозначим за u и v . Переберем всевозможные варианты того, какие расстояния до них от заданной точки могли быть. Пусть эти расстояния равны соответственно d_1 и d_2 . Заметим, что искомая вершина лежит на расстоянии d_1 от вершины u тогда и только тогда, когда эта вершина лежит на окружности радиуса d_1 в центре u . Аналогично и для v, d_2 - вершина лежит на окружности радиуса d_2 в центре v . То есть потенциальная вершина - точка пересечения двух окружностей. Переберем всевозможные $O(n^2)$ вариантов пар расстояний, для каждого случая найдем точки пересечения окружностей (их

будет не больше 2), а затем за $O(n \log n)$ для каждой потенциальной точки проверим, подходит ли она (время на сортировку).

Асимптотика: $O(n^3 \log n)$

Подзадача 3

В решении данной подзадачи воспользуемся замечаниями из решения второй подзадачи. Мы можем перефразировать нашу задачу так: необходимо распределить расстояния от искомой вершины (a, b) до данных среди них так, чтобы имелось пересечение полученных окружностей. Если записать данное условие как систему, получим:

$$\begin{cases} (x_1 - a)^2 + (y_1 - b)^2 = d_1^2 \\ (x_2 - a)^2 + (y_2 - b)^2 = d_2^2 \\ \dots \\ (x_n - a)^2 + (y_n - b)^2 = d_n^2 \end{cases}$$

Давайте немного преобразуем наши равенства. Сложим их:

$$\sum_{i=1}^n d_i = \sum_{i=1}^n ((x_i - a)^2 + (y_i - b)^2)$$

$$\sum_{i=1}^n d_i = \sum_{i=1}^n (x_i^2 - 2x_i a + a^2 + y_i^2 - 2y_i b + b^2)$$

$$\sum_{i=1}^n d_i = \sum_{i=1}^n x_i^2 - 2a \sum_{i=1}^n x_i + n \cdot a^2 + \sum_{i=1}^n y_i^2 - 2b \sum_{i=1}^n y_i + n \cdot b^2$$

$$\sum_{i=1}^n d_i - \sum_{i=1}^n x_i^2 - \sum_{i=1}^n y_i^2 = -2a \sum_{i=1}^n x_i + n \cdot a^2 - 2b \sum_{i=1}^n y_i + n \cdot b^2$$

$$\frac{\sum_{i=1}^n d_i - \sum_{i=1}^n x_i^2 - \sum_{i=1}^n y_i^2}{n} = -\frac{2a \sum_{i=1}^n x_i}{n} + a^2 - \frac{2b \sum_{i=1}^n y_i}{n} + b^2$$

$$\begin{aligned} \frac{\sum_{i=1}^n d_i - \sum_{i=1}^n x_i^2 - \sum_{i=1}^n y_i^2}{n} + \left(\frac{\sum_{i=1}^n x_i}{n} \right)^2 + \left(\frac{\sum_{i=1}^n y_i}{n} \right)^2 = \\ = -\frac{2a \sum_{i=1}^n x_i}{n} + a^2 - \frac{2b \sum_{i=1}^n y_i}{n} + b^2 + \left(\frac{\sum_{i=1}^n x_i}{n} \right)^2 + \left(\frac{\sum_{i=1}^n y_i}{n} \right)^2 \end{aligned}$$

Справа выносятся два полных квадрата, получаем:

$$\frac{\sum_{i=1}^n d_i - \sum_{i=1}^n x_i^2 - \sum_{i=1}^n y_i^2}{n} + \left(\frac{\sum_{i=1}^n x_i}{n} \right)^2 + \left(\frac{\sum_{i=1}^n y_i}{n} \right)^2 = \left(a - \left(\frac{\sum_{i=1}^n x_i}{n} \right) \right)^2 + \left(b - \left(\frac{\sum_{i=1}^n y_i}{n} \right) \right)^2$$

Что-то напоминает формула? А если я запишу так: $r^2 = (x - x_0)^2 + (y - y_0)^2$?

Все верно. Мы получили формулу окружности с квадратом радиуса, которые равен левой части последнего выражения, а его центр равен среднему значению по каждой из координат. И это значит, что искомая точка принадлежит данной окружности.

Теперь, как в подзадаче 2, переберем для первой точки u , какое будет от нее расстояние до искомой, аналогично получим задачу поиска пересечения двух окружностей (только теперь

вторая окружность задается выше указанным способом), рассмотрим каждую потенциальную точку и найдем ответ.

Самое интересно - асимптотика. Изначально, смотря на данный алгоритм, легко оценить асимптотику как $O(n^2 \log n)$, поскольку $O(n)$ потенциальных точек, для каждой проверка за $O(n \log n)$. Но на самом деле, асимптотика данного решения гораздо меньше, поскольку наши входные данные на точки случайны (как раз здесь и нужно условие равновероятной генерации точек), вероятность того, что пересечение двух таких окружностей еще и целым окажется (поскольку загаданная точка целочисленна) невероятно мала, потенциальных точек будет очень мало. Поэтому, считая, что точек будет в среднем $O(1)$ на каждом тесте, получаем итоговую асимптотику $O(n \log n)$

Асимптотика: $O(n \log n)$