

## Задача А. Корм для мальтипу

Каждой собаке необходимо не менее  $C$  грамм корма. Всего собак  $N$ , значит общее минимальное количество корма, которое нужно, равно  $N \cdot C$  грамм.

Одна пачка содержит  $W$  грамм корма. Нам нужно найти минимальное количество пачек, суммарного объёма которых хватит, чтобы покрыть требуемое количество корма.

Обозначим ответ как  $Ans$ . Тогда должно выполняться:  $Ans \cdot W \geq N \cdot C$ .

Отсюда:  $Ans \geq \frac{N \cdot C}{W}$ .

Так как количество пачек — целое число, берём округление вверх:  $Ans = \lceil \frac{N \cdot C}{W} \rceil$ .

Это и есть минимальное необходимое количество пачек корма.

Сложность решения:  $O(1)$  по времени и  $O(1)$  по памяти.

## Задача В. Парковка

В первой подзадаче ответ это  $\frac{A \cdot (A-1)}{2}$  так как лимузины можно поставить лесенкой и останется чуть меньше половины паркомест

Во второй ( $A \geq K$ ) ответ это  $A \cdot A - \frac{K \cdot (K+1)}{2}$ , так как суммарный размер лимузинов это  $\frac{K \cdot (K+1)}{2}$  и их снова можно поставить лесенкой

В третьей подзадаче при  $A = K$  ответ как в первой подзадаче, а  $A < K$  ответ очевидно "Impossible"

В четвертной надо заметить, что если  $A > K$ , то самый большой лимузин не помещается по длине, так что ответ "Impossible" а если  $A \leq K$ , то ответ как во второй подзадаче

## Задача С. Люттик и дартс

Пусть было сделано  $a$  бросков в область 5,  $b$  бросков в область 3 и  $c$  бросков в область 1. Тогда  $5a + 3b + c = x$  и  $a + b + c = k$ , а значит  $4a + 2b = x - k$ , следовательно  $2a + b = \frac{x-k}{2}$ . Пусть  $t = \frac{x-k}{2}$ , тогда  $2a + b = t$ . Отсюда  $b = t - 2a$ . Поскольку  $0 \leq b$  и  $a + b \leq k$ , получаем два неравенства:

$$0 \leq b \Rightarrow 0 \leq t - 2a \Rightarrow a \leq \frac{t}{2}$$

$$a + b \leq k \Rightarrow a + (t - 2a) \leq k \Rightarrow t - a \leq k \Rightarrow a \geq t - k$$

Также мы знаем, что  $a \geq 0$ .

Таким образом, мы получили ограничения  $\max(0, t - k) \leq a \leq \lfloor \frac{t}{2} \rfloor$ . Значит, минимальное  $a$  имеет вид:  $a_{min} = \max(0, t - k) = \max(0, \frac{x-k}{2} - k) = \max(0, \frac{x-3k}{2})$ . Используя это значение, найдем  $b = \frac{x-k}{2} - 2a$  и  $c = k - a - b$ .

В ходе решения мы опирались на следующие ограничения:

$$x \geq k$$

$$x - k - \text{чётное}$$

$$x \leq 5k$$

$$\max(0, t - k) \leq \lfloor \frac{t}{2} \rfloor$$

## Задача D. Жимовик

Разбор

Умножим все веса на 2. Тогда блины имеют веса 5, 10, 20, 40, 80, а для подхода с весом  $a_i$  на штанге на каждой стороне нужно собрать вес  $a_i/2$ , то есть в новых единицах это сумма  $a_i/5$ .

Если  $a_i$  не делится на 5, то ответ невозможен, потому что любой собранный вес кратен 5.

Дальше обозначим  $x_i = a_i/5$ . Тогда на одной стороне доступны веса 1, 2, 4, 8, 16. Это степени двойки, поэтому минимальный набор блинов для числа  $x_i$  единственен и задаётся двоичной записью числа  $x_i$ . Если в двоичной записи на некотором месте стоит 1, то соответствующий блин входит в набор, иначе не входит.

Пусть для числа  $x_i$  получена последовательность блинов на одной стороне, записанная по возрастанию весов. Например, для  $x_i = 13$  это последовательность 1, 4, 8.

Теперь рассмотрим переход от подхода  $a_i$  к подходу  $a_{i+1}$ . Пусть на одной стороне штанги получены последовательности  $s_i$  и  $s_{i+1}$ . Так как блины на штанге лежат от грифа к краю в порядке возрастания, сначала можно оставить только общий префикс этих двух последовательностей. Все, что стоит после этого префикса в старой конфигурации, нужно снять, а все, что стоит после него в новой, нужно надеть.

Пусть  $p$  — длина наибольшего общего префикса последовательностей  $s_i$  и  $s_{i+1}$ . Тогда на одной стороне число операций равно  $|s_i| + |s_{i+1}| - 2p$ . Так как левая и правая стороны одинаковые, итоговая стоимость перехода равна  $2 \cdot (|s_i| + |s_{i+1}| - 2p)$ .

Для первого подхода считаем, что штанга пустая, то есть  $s_0$  — пустая последовательность.

Таким образом, для каждого  $a_i$  строим его минимальную последовательность блинов, а затем суммируем стоимости переходов между соседними подходами.

Подгруппы

Группа 1 ( $n = 1, 1 \leq a_i \leq 80$ )

Здесь можно просто перебрать все варианты для одной стороны. Всего существует только  $2^5 = 32$  подмножеств блинов, поэтому ответ легко находится полным перебором или рекурсией по пяти типам блинов.

Группа 2 ( $n = 1, 1 \leq a_i \leq 400$ )

Тоже можно использовать перебор всех наборов блинов для одной стороны, но уже удобнее сразу строить минимальную конфигурацию по двоичной записи числа  $x = a_i/5$ .

Группа 3 ( $n = 1$ )

Для одного подхода строится единственная минимальная конфигурация по двоичной записи числа  $x = a_i/5$ . Если  $a_i$  не делится на 5, ответ невозможен.

Группа 4 ( $1 \leq a_i \leq 80$ )

Для каждого подхода строим минимальную конфигурацию независимо. Переходы между соседними подходами считаем по длине общего префикса.

Группа 5 ( $1 \leq a_i \leq 400$ )

Используем ту же идею: для каждого  $a_i$  строим последовательность блинов по двоичной записи числа  $x = a_i/5$ , затем считаем стоимость переходов между соседними конфигурациями.

Группа 6 (без ограничений)

Для каждого  $a_i$  за  $O(1)$  строим последовательность блинов на одной стороне, после чего находим длину общего префикса с предыдущей последовательностью и добавляем стоимость перехода.

Сложность

На построение одной конфигурации уходит  $O(1)$ , потому что типов блинов всего 5.

Каждый переход тоже считается за  $O(1)$ .

Итоговая сложность решения:  $O(n)$

## Задача Е. Метро

В первой подзадаче можно было просто перебрать откуда и куда может быть проведен перегон и посчитать в каждом случае ответ.

Во второй подзадаче надо было заметить, что перегон нужно строить между центральной станцией и какой-нибудь другой, так как в противном случае можно заменить конец перегона, который лежит ближе к центральной станции на нее и тогда расстояние до второго конца перегона уменьшится, следовательно и суммарное уменьшится. Таким образом можно получить решения за  $O(n^2)$

В третьей подзадаче можно было сделать например тернарный поиск и вывести ответ формулой. Для этого нужно заметить, что если построить перегон от станции 1 до станции  $l$ , то расстояние до всех станций от  $l+1$  до  $n$  будет  $2, 3, \dots$ , а так же изменится расстояние до всех станций с номерами от  $\frac{l+3}{2}$  до  $l$ , так как они ближе к  $l$  чем к 1.

В четвертой подзадаче нужно было посмотреть на метро как на дерево и написать снова решение с перебором, но теперь нужно применить идею из прошлой подзадачи и перебирать только вершины на пути из корня в текущую и проверять, правда ли они уже дальше от корня, чем от текущей вершины (в которую проводим ребро). Если дальше, то расстояние до нее и до всего ее поддерева изменяется. Новую сумму можно посчитать формулой для случая бинарных деревьев или предсчитать. И того получим решение за  $O(n \log n)$

В подзадаче 5 будем рассматривать только рёбра  $(v, u)$ , где  $v$  — предок  $u$ . При добавлении ребра расстояния меняются:

- Для вершин поддерева  $u$  сокращение равно  $d[u] - d[v] - 1$ , выгода  $sz[u](d[u] - d[v] - 1)$ .

- Для вершин  $w$  на пути от  $v$  к  $u$ , лежащих вне поддерева  $u$ , сокращение возможно при  $2d[w] > d[u] + d[v] + 1$  и равно  $2d[w] - d[u] - d[v] - 1$  для самой  $w$  и всех ветвей, не содержащих  $u$ .

Итоговую сумму можно вычислить, если для текущей вершины  $u$  перебирать предка  $v$  и знать суммы размеров и расстояний для частей дерева, отрезанных от пути  $(v, u)$ .

**Алгоритм за  $O(n)$ .** Запустим DFS, поддерживая стек  $st$  компонент (поддеревьев, отходящих от текущего пути). Элемент стека  $i$  (нумерация с 0) соответствует вершине глубины  $i$  и хранит  $sz_i$ ,  $sum_i$  (размер и сумму расстояний внутри компоненты). Пусть  $s = \sum_{i \geq ndx} sum_i$ ,  $sz_1 = \sum_{i \geq ndx} sz_i$ ,  $sz_k = \sum_{i \geq ndx} sz_i \cdot i$ , а  $ans_1$  — уже оптимизированное расстояние для компонент  $i < ndx$ . Индекс  $ndx$  (центр) сдвигается при нечётной длине стека (или при длине 2). Тогда для текущей вершины  $u$  и выбранного  $v$  (соответствует  $ndx$ ) сумма всех расстояний равна

$$ans = ans_1 + s + sz_1 \cdot (|st|) - sz_k.$$

Перебирая  $u$  и обновляя стек при спуске/подъёме, находим минимум  $ans$ .

Общее время работы  $O(n)$ , память  $O(n)$ .

## Задача F. Олимпиада 2077

### Общее наблюдение

Нужно выбрать целые неотрицательные  $x, y, z$  такие, что

$$x + y + z = k.$$

Группа  $i$  принесёт  $d_i$  дипломов, если

$$x \geq a_i, \quad y \geq b_i, \quad z \geq c_i.$$

Удобно заменить равенство на условие

$$x + y + z \leq k.$$

Если сумма меньше  $k$ , оставшиеся баллы можно добавить в любую координату. Все условия имеют вид «координата не меньше порога», поэтому ответ от этого не уменьшится.

Сумму дипломов нужно хранить в 64-битном типе.

### Подгруппы 1–3, $t = 1$

Здесь для всех  $i$  выполнено

$$c_i = 0.$$

Третья координата не влияет на прохождение групп. Нужно выбрать  $x$  и  $y$  так, чтобы

$$x + y \leq k,$$

и максимизировать сумму  $d_i$  по группам, для которых

$$a_i \leq x, \quad b_i \leq y.$$

Достаточно рассматривать  $x$  из множества  $\{0\} \cup \{a_i\}$ . Отсортируем группы по  $a_i$  и будем идти по  $x$  по возрастанию.

Когда становится выполнено  $a_i \leq x$ , добавим группу в дерево Фенвика по координате  $b_i$  с весом  $d_i$ .

Для текущего  $x$  максимально возможное значение  $y$  равно

$$y = k - x.$$

Значит нужно узнать сумму всех уже добавленных групп с

$$b_i \leq k - x.$$

Это префиксный запрос в дереве Фенвика.

Сложность:

$$O(n \log n).$$

**Подгруппы 4–6,  $t = 2$**

Здесь для всех  $i$  выполнено

$$c_i = b_i.$$

Группа проходит, если

$$a_i \leq x, \quad b_i \leq y, \quad b_i \leq z.$$

Обозначим

$$q = \min(y, z).$$

Тогда группа проходит, если

$$a_i \leq x, \quad b_i \leq q.$$

Чтобы было  $\min(y, z) \geq q$ , нужно потратить хотя бы  $2q$  баллов на координаты  $y$  и  $z$ . Поэтому ограничение становится таким:

$$x + 2q \leq k.$$

Теперь задача снова двумерная.

Перебираем  $x$  по возрастанию среди  $\{0\} \cup \{a_i\}$ . Все группы с  $a_i \leq x$  добавляем в дерево Фенвика по координате  $b_i$ .

Для текущего  $x$  максимальное возможное  $q$  равно

$$q = \left\lfloor \frac{k - x}{2} \right\rfloor.$$

Запрашиваем сумму активных групп с

$$b_i \leq \left\lfloor \frac{k - x}{2} \right\rfloor.$$

Сложность:

$$O(n \log n).$$

**Подгруппы 7–9,  $t = 3$**

Здесь выполнено условие

$$c_i \geq b_i \geq c_{i-1}.$$

Из него следует, что последовательности  $b_i$  и  $c_i$  не убывают.

Действительно, так как  $c_{i-1} \geq b_{i-1}$  и  $b_i \geq c_{i-1}$ , то

$$b_i \geq b_{i-1}.$$

Также из  $c_i \geq b_i \geq c_{i-1}$  следует

$$c_i \geq c_{i-1}.$$

Значит если по координатам  $y$  и  $z$  проходит некоторая группа  $p$ , то все предыдущие группы тоже проходят по этим двум координатам. Поэтому выбор  $y$  и  $z$  задаёт некоторый префикс групп.

Переберём последнюю группу  $p$  в этом префиксе.

Чтобы покрыть по координатам  $y$  и  $z$  все группы  $1, 2, \dots, p$ , достаточно взять

$$y = b_p, \quad z = c_p.$$

Тогда на координату  $x$  остаётся

$$x = k - b_p - c_p.$$

Среди первых  $p$  групп засчитываются те, для которых

$$a_i \leq k - b_p - c_p.$$

Идём по  $p$  слева направо. При переходе к очередному  $p$  добавляем группу  $p$  в дерево Фенвика по координате  $a_i$  с весом  $d_i$ .

После добавления делаем префиксный запрос по всем

$$a_i \leq k - b_p - c_p.$$

Максимум по всем  $p$  даёт ответ.

Сложность:

$$O(n \log n).$$

#### Подгруппы 10–11, $t = 4$

Здесь дополнительных ограничений нет. Полное решение при максимальных ограничениях не требуется, а в нужных подгруппах достаточно  $n \leq 5000$ .

Решим за

$$O(n^2).$$

Достаточно рассматривать значения

$$x \in \{0\} \cup \{a_i\}.$$

Если оптимальное  $x$  лежит между двумя соседними такими значениями, его можно уменьшить до ближайшего меньшего значения. Набор групп с  $a_i \leq x$  не изменится, а свободный запас для  $y + z$  только увеличится.

Аналогично достаточно рассматривать

$$y \in \{0\} \cup \{b_i\}.$$

Зафиксируем  $x$ .

Группа активна по первой координате, если

$$a_i \leq x.$$

Для активной группы условия по  $y$  и  $z$  такие:

$$b_i \leq y, \quad c_i \leq k - x - y.$$

Второе условие перепишем:

$$y \leq k - x - c_i.$$

Значит при фиксированном  $x$  каждая активная группа добавляет вес  $d_i$  на отрезок значений  $y$ :

$$b_i \leq y \leq k - x - c_i.$$

Теперь нужно найти точку  $y$ , покрытую отрезками с максимальной суммой весов.

Сожмём все значения из множества  $\{0\} \cup \{b_i\}$ . Для каждого фиксированного  $x$  заведём разностный массив по этим значениям.

Левая граница отрезка для группы  $i$  известна сразу: это позиция значения  $b_i$ .

Правую границу нужно найти для значения

$$k - x - c_i.$$

Чтобы не делать бинарный поиск, заранее отсортируем группы по убыванию  $c_i$ .

Для фиксированного  $x$  будем идти по группам в этом порядке. Тогда величина

$$k - x - c_i$$

не убывает, а значит указатель правой границы по массиву сжатых значений  $y$  только движется вправо.

Для каждой группы, если  $a_i \leq x$ , добавляем  $d_i$  на соответствующий отрезок в разностном массиве. Если левая граница оказалась правее правой, такая группа не добавляется.

После обработки всех групп считаем префиксные суммы разностного массива и обновляем ответ.

Для одного фиксированного  $x$  все действия занимают

$$O(n).$$

Всего значений  $x$  не больше  $n + 1$ , поэтому итоговая сложность:

$$O(n^2).$$

Память:

$$O(n).$$

### Итоговые сложности

Для  $t = 1$ :

$$O(n \log n).$$

Для  $t = 2$ :

$$O(n \log n).$$

Для  $t = 3$ :

$$O(n \log n).$$

Для  $t = 4$  в требуемых подгруппах:

$$O(n^2).$$

## Задача G. Инверсии в графе

### Анализ обходов и структуры данных

Для начала проанализируем свойства прямого (**pre-order**) и обратного (**post-order**) обходов в дереве поиска в глубину (DFS). Алгоритм, описанный в условии, строит стандартное дерево DFS, выбирая на каждом шаге ребро в вершину с минимальным номером. Это гарантирует единственность дерева для фиксированного корня (в данной задаче подразумевается запуск из вершины 1).

Пусть для любых двух вершин  $u$  и  $v$  выполняется отношение  $u$  является предком  $v$  в дереве DFS. Тогда:

1. В прямом обходе вершина  $u$  будет посещена раньше  $v$ , так как алгоритм сначала заходит в корень поддерева:  $\text{prePos}[u] < \text{prePos}[v]$ .
2. В обратном обходе вершина  $u$  будет добавлена в массив только после того, как все её потомки (включая  $v$ ) будут полностью обработаны:  $\text{postPos}[u] > \text{postPos}[v]$ .

Таким образом, если одна вершина является предком другой, их относительный порядок в массивах  $\text{pre}$  и  $\text{post}$  **различается**.

#### Случай несвязанных вершин (разные поддеревья)

Рассмотрим случай, когда ни  $u$  не является предком  $v$ , ни  $v$  не является предком  $u$ . Это происходит, когда при обходе из некоторого общего предка алгоритм сначала полностью завершил обход ветви, содержащей одну вершину, и только потом перешёл в ветвь, содержащую другую.

1. Пусть ветвь с вершиной  $u$  была посещена раньше ветви с вершиной  $v$ .
2. Тогда  $\text{prePos}[u] < \text{prePos}[v]$ .
3. Так как обход поддерева  $u$  завершится до того, как алгоритм доберётся до  $v$ , то и в массив  $\text{post}$  вершина  $u$  попадёт раньше:  $\text{postPos}[u] < \text{postPos}[v]$ .

Следовательно, условие задачи (совпадение относительного порядка) выполняется тогда и только тогда, когда между вершинами  $u$  и  $v$  **нет отношения «предок — потомок»** в дереве DFS.

#### Алгоритм решения

Для решения задачи нам необходимо:

1. Построить дерево DFS. Для этого для каждой вершины нужно отсортировать список её соседей по возрастанию, чтобы соблюсти условие выбора вершины с минимальным номером.
2. Запустить DFS из вершины 1. Во время обхода для каждой вершины  $v$  нужно вычислить размер её поддерева  $\text{sz}[v]$  (количество вершин в поддереве, включая саму  $v$ ).
3. Общее количество пар вершин  $(v, u)$  при  $1 \leq v < u \leq n$  равно  $\frac{n(n-1)}{2}$ .
4. Количество пар, в которых одна вершина является предком другой, равно  $\sum_{v=1}^n (\text{sz}[v] - 1)$ . Каждая такая пара будет иметь разный порядок в  $\text{pre}$  и  $\text{post}$ .
5. Искомый ответ — это разность общего количества пар и количества пар с отношением предок-потомок.

#### Математическая формулировка

Итоговая формула для вычисления количества подходящих пар:

$$\text{Ответ} = \frac{n(n-1)}{2} - \sum_{v=1}^n (\text{sz}[v] - 1)$$

#### Сложность алгоритма

1. Сортировка списков смежности:  $O(m \log m)$  или  $O(m \log n)$ .
2. Построение дерева DFS и подсчёт размеров поддеревьев:  $O(n + m)$ .
3. Итоговая временная сложность:  $O(m \log n)$ , что укладывается в ограничения при  $n, m \leq 2 \cdot 10^5$ .
4. Память:  $O(n + m)$  для хранения графа и массивов.