Задача A. Круг из камней

Пусть W и B — количество белых и чёрных камней соответственно. Операция 1 меняет k белых и $k+1$ чёрных камней на 1 чёрный; аналогично, операция 2 меняет $k+1$ белых и k чёрных камней на 1 белый. Таким образом, значение $W - B$ является инвариантом; поскольку в конце оно должно быть равно 0, изначально должно выполняться $W = B$.

Покажем, что это достаточное условие. Рассмотрим какой-нибудь чёрный камень, тогда на отрезке, содержащем все камни, кроме него, W белых и $B - 1$ чёрных камней, то есть его можно заменить на один белый камень.

Таким образом, ответ —«`Yes`» тогда и только тогда, когда $W = B$. Асимптотика: $O(|s|)$.

Задача B. Составные числа

Все простые числа, кроме 3, имеют вид $3k \pm 1$. Отсюда квадраты всех простых чисел, кроме 3, имеют вид $3k + 1$. Таким образом, если $x = 3k + 2$, то $p^2 + x$ делится на 3 для всех простых $p > 3$. Чтобы $p^2 + x$ было составным для всех нечётных простых, достаточно потребовать, чтобы $x \bmod 3 = 2$ и $x + 9$ было составным.

Утверждается, что такие числа встречаются часто, поэтому можно либо перебрать несколько чисел, начиная с n, проверяя $x + 9$ на простоту за $O(\sqrt{n})$, либо заметить, что числа вида $10^k + 16$ удовлетворяют условию при $k > 0$.

Задача C. Результирующая стопка

Очевидно, что в каждый момент времени нам выгодно добавить в стопку один из минимальных возможных элементов. Заметим, что если есть два минимальных элемента, но элементы под ними различаются, мы можем взять тот, под которым элемент меньше. Действительно, если больший элемент в ответе идёт раньше меньшего, ответ неоптимален. Попробуем обобщить это наблюдение.

Утверждение: если каждый раз брать элемент из лексикографически минимальной последовательности из оставшихся (причём если одна является префиксом второй, то вторая предпочтительнее), то мы получим оптимальный ответ.

Назовём последовательность из утверждения P. Пусть в оптимальном ответе мы взяли элемент из какой-то другой последовательности Q.

Пусть существует позиция i, в которой P и Q впервые различаются, тогда $P_i < Q_i$. Построим другой ответ следующим образом: каждый раз, когда мы берём элемент из P или Q, мы будем брать элемент из другой последовательности, либо пока не возникнет ситуация, что мы взяли одинаковое количество элементов из P и Q, либо пока не возьмём P_i или Q_i.

Пусть в исходном ответе мы взяли Q_i раньше. Тогда, поскольку до i-го элемента P и Q совпадают, а вместо Q_i в новом ответе будет P_i, новый ответ будет строго лучше. Если же в исходном ответе мы взяли P_i раньше, то это означает, что в какой-то этот момент времени мы взяли больше элементов из P, чем из Q. Следовательно, мы перестанем инвертировать наши действия до того, как возьмём P_i, и получим в точности такой же ответ.

Если Q — префикс P, аналогичная конструкция позволяет всегда построить ответ, равный исходному. Таким образом, мы показали, что существует оптимальный ответ, начинающийся с P_1.

Пользуясь таким жадным алгоритмом, мы можем построить ответ, если будем уметь быстро сравнивать суффиксы последовательностей. Существует несколько известных способов это сделать, например, бинарный поиск с хешами ($O(nk)$ на построение и $O(\log k)$ на запрос) и суффиксный массив ($O(nk \log^2 k)$ или $O(nk \log k)$ на построение и $O(\log k)$ или $O(1)$ на запрос). После этого можно поддерживать минимальную из оставшихся последовательностей в очереди с приоритетом.

Асимптотика решения: $O(nk \log n \log k)$ или $O(nk \log nk)$ в зависимости от выбранного алгоритма. Из-за больших ограничений ($\sum k \leqslant 10^6$) требуется аккуратность в выборе и реализации алгоритма.

Задача D. Дождь над Долгопрудным

Рассмотрим улицы как непересекающиеся отрезки на осях Ox и Oy.

Рассмотрим произвольный прямоугольник $w \times h$ со сторонами, параллельными осям координат. Пусть L_x — это длина пересечения отрезков на Ox, задающих улицы, с проекцией прямоугольника на Ox, L_y — аналогичная длина на Oy. Тогда площадь пересечения прямоугольника с улицами равна $L_x \cdot h + L_y \cdot w - L_x \cdot L_y$ по формуле включений-исключений.

Чтобы вычислить L_x и L_y, можно бинарным поиском найти самый левый и самый правый отрезки, полностью лежащие в соответствующей проекции, после чего прибавить длину пересечения с соседними слева и справа отрезками.

На i-й день туча будет занимать прямоугольник с углами $(i \cdot u, i \cdot v)$ и $(w + i \cdot u, h + i \cdot v)$. Переберём i; для каждого будем находить площадь пересечения с улицами, как показано выше. Асимптотика: $O(t \log nm)$.

Задача E. Вирус поразил компьютер

Чтобы найти ответ, нам достаточно знать, сколько раз каждый отрезок $[l; r]$ будет встречаться в памяти через d дней. Действительно, ответ — это сумма $\#[l; r] \cdot (a_l + \cdots + a_r)$ по всем $[l; r]$.

Введём $\mathrm{dp}[i][l][r]$ — сколько раз отрезок $[l; r]$ будет встречаться в памяти после i дней. $\mathrm{dp}[0][1][n] = 1$ и $\mathrm{dp}[0][l][r] = 0$ для любого другого $[l; r]$.

Рассмотрим пересчёт $\mathrm{dp}[i+1]$ из $\mathrm{dp}[i]$. Наивный алгоритм прибавляет $\mathrm{dp}[i][l][r]$ к $\mathrm{dp}[i+1][l'][r']$ для любых $l \leqslant l' \leqslant r' \leqslant r$. Такой алгоритм работает за $O(n^4)$, что недостаточно быстро.

Допустим, что $\mathrm{dp}[i][l][r] = 0$ для $l > r$. Тогда, если мы представим значения $\mathrm{dp}[i]$ как матрицу $n \times n$, $\mathrm{dp}[i+1][l][r]$ — это сумма $\mathrm{dp}[i][x][y]$ на подпрямоугольнике $l \leqslant x \leqslant r$; $l \leqslant y \leqslant r$. Найти её можно с помощью префиксных сумм за $O(1)$. Таким образом, мы получаем пересчёт за $O(n^2)$.

Итоговая асимптотика: $O(n^2 d)$ на набор входных данных.

Отметим, что существует множество альтернативных, а также более эффективных решений задачи.

Задача F. Новая цифровая клавиатура

Заметим, что в любой момент времени нас интересует только то, на какой клавише мы находимся и какой остаток по модулю m даёт уже набранное число.

Введём $\mathrm{dp}[i][x]$ — минимальное количество операций, после которых мы находимся на клавише с цифрой i и набранное число даёт остаток x по модулю m. Из этого состояния мы можем перейти в (j, x), если клавиши с цифрами i и j находятся рядом на клавиатуре, а также в $(i, (10x + i) \bmod m)$.

Ответ на задачу — минимум из всех $\mathrm{dp}[i][r]$. Поскольку каждый переход занимает ровно одну операцию, найти все $\mathrm{dp}[i][x]$ можно поиском в ширину. Асимптотика: $O(m)$.

Задача G. Задачка на разминку

Пусть k — старший бит в y, тогда $y = 2^k + r$, $r < 2^k$. Если $x < 2^k$, то $y \bmod x < x < 2^k$, но $y \oplus x = 2^k + (r \oplus x)$.

Отсюда получаем, что старшие биты в y и x совпадают. Но это означает, что $y < 2x$, следовательно, $y \bmod x = y - x$. Утверждается, что это условие эквивалентно тому, что при вычитании x из y мы не занимаем единицу из старших разрядов, то есть что x — подмаска y. Действительно, если мы занимаем единицу из i-го разряда, но не занимаем единицу из $i+1$-го разряда, i-й бит $y - x$ не будет равен исключающему ИЛИ i-ых битов x и y.

Таким образом, нам необходимо посчитать количество таких пар (x, y), что $a \leqslant x, y \leqslant b$, $y \& x = x$ и старшие биты y и x совпадают.

Рассмотрим $\mathrm{dp}[i][\mathrm{flag}_a][\mathrm{flag}_b][\mathrm{flag}_{\mathrm{MSB}}]$ — количество способов зафиксировать младшие i бит x и y, если из старших бит уже известно, что:

- flag_a — (x гарантированно больше a) либо (все биты старше i в x и a совпадают)
- flag_b — (y гарантированно меньше b) либо (все биты старше i в y и b совпадают)
- $\mathrm{flag}_{\mathrm{MSB}}$ — (старшие биты x и y совпадают) либо (все биты старше i в x и y нулевые)

Каждое состояние можно пересчитать, сделав i-ые биты x и y равными 0 и 0, 0 и 1 или 1 и 1 соответственно, причём флаги могут запрещать некоторые переходы. Например, если i-ый бит a равен 1 и из flag_a известно, что все старшие биты x и a совпадают, возможен только третий переход. Асимптотика данного решения: $O(\log C)$.

Отметим, что, поскольку $\log C \leqslant 60$, допустимы менее эффективные решения задачи.

Задача H. Постройка общежитий

Утверждение: как бы мы ни расположили прямоугольники, найдётся либо вертикальная, либо горизонтальная прямая такая, что она не пересекает прямоугольники и что все три прямоугольника не лежат по одну сторону от неё.

Доказательство: если проекция прямоугольника на Ox пересекается с проекцией другого прямоугольника, то их проекции на Oy не пересекаются; действительно, иначе пересекаются сами прямоугольники. Пусть не существует вертикальной разделяющей прямой, тогда для одного из прямоугольников проекция на Ox пересекается с обеими проекциями других прямоугольников. Тогда прямая, содержащая либо верхнюю, либо нижнюю сторону этого прямоугольника, будет разделяющей.

Поскольку мы можем поворачивать прямоугольники, без ограничения общности предположим, что разделяющая прямая — вертикальная.

Зафиксируем ориентацию прямоугольников. Переберём прямоугольник, лежащий по другую сторону прямой от двух других. Пусть его размер — $h_1 \times w_1$, а размеры двух других — $h_2 \times w_2$ и $h_3 \times w_3$.

Минимальный прямоугольник, в который помещаются второй и третий — это либо $\max(h_2, h_3) \times (w_2 + w_3)$, либо $(h_2 + h_3) \times \max(w_2, w_3)$. Добавляя первый прямоугольник к этим, получаем два варианта ответа: $\max(h_1, h_2, h_3) \times (w_1 + w_2 + w_3)$ и $\max(h_1, h_2 + h_3) \times (w_1 + \max(w_2, w_3))$.

Всего получаем $2^3 \cdot 3 \cdot 2 = 48$ возможных ответов.

Задача I. Заработайте на массиве

Утверждается, что существует оптимальный ответ, в котором все элементы массива равны какому-то из c_i. Действительно, если это не так, то элементы можно увеличить до ближайшего c_i, и прибыль не уменьшится, поскольку отрезки, для которых соответствующий элемент является минимумом, начнут приносить больше денег.

Рассмотрим минимальный элемент, равный p. Все отрезки, содержащие его, принесут либо p денег, либо нисколько. Если мы переберём позицию и величину первого минимума, мы можем свести задачу к двум меньшим — на соответствующем префиксе и суффиксе, причём на каждом из них можно рассматривать только те отрезки, которые полностью лежат в рассматриваемом.

Пусть $d_1, d_2, \ldots, d_k$ — все возможные значения элементов массива. Рассмотрим $\mathrm{dp}[i][l][r]$ — максимальная возможная гарантированная прибыль с отрезка $[l; r]$, если все значения на нём не меньше d_i. Пересчитать это состояние можно, выбрав позицию m и величину $d_j \geqslant d_i$ первого минимума, через $\mathrm{dp}[j+1][l][m-1] + \mathrm{dp}[j][m+1][r] + d_j \cdot \mathrm{cnt}$, где cnt — это количество отрезков, лежащих в $[l; r]$ и содержащих m, таких что $c_k \geqslant d_j$.

Покажем, как делать эффективный пересчёт. Зафиксируем текущий отрезок $[l; r]$ и переберём m. Найдём все отрезки, лежащие в $[l; r]$ и содержащие m, и отсортируем их по c. Переберём d_j; в каждый момент времени cnt равно длине суффикса отрезков с $c_k \geqslant d_j$. Каждое значение d_j пересчитывает $\mathrm{dp}[i][l][r]$ для $i \leqslant j$.

Итого получаем $O(mn^3)$ на пересчёт, если использовать сортировку подсчётом и подерживать длину суффикса указателем на его первый элемент. Восстановить ответ можно рекурсивно, если запомнить для каждого состояния оптимальную пару (m, j).

Задача J. Датацентр

Рассмотрим любую расстановку диагональных переходов, обладающую следующим «оптимальным» свойством: если представить исходные компоненты как вершины в графе, а добавленные переходы — как соединяющие их рёбра, то между любыми двумя вершинами существует не более одного простого пути (то есть граф является лесом).

Пусть такая расстановка не удовлетворяет условию задачи, то есть существует подмножество клеток одной компании, не соединённое с остальными.

Если хотя бы одно из таких подмножеств не примыкает к границе, то оно со всех сторон окружено клетками другой компании. Если из какой-либо клетки подмножества ведёт наружу какой-либо неиспользованный переход, достаточно просто добавить его. Иначе все диагональные переходы на границе подмножества принадлежат другой компании. Но тогда они образуют цикл, что противоречит свойству выше.

Пусть теперь все такие подмножества примыкают к границе (при этом их хотя бы два). Рассмотрим все клетки, примыкающие к границе; они образововыют объединённые подотрезки, причём либо все эти клетки принадлежат одной компании, либо они чередуются.

Предположим, что на границе есть хотя бы четыре последовательных подотрезка A, B, C и D, причём A и C принадлежат одной компании, а B и D — другой. Если выполнить условие задачи возможно, то A и C должны быть соединены некоторым путём из непосредственных и добавленных диагональных переходов. Однако поскольку A и C примыкают к границе, этот путь делит всю матрицу на две части, которые не могут быть соединены переходами другой компании; при этом B и D лежат в разных частях.

Следовательно, каждой компании принадлежит не более одного граничного отрезка. Но тогда ситуация с двумя несвязанными подмножествами, примыкающими к границе, невозможна.

Отсюда получаем решение: переберём все возможные диагональные переходы, добавляя их, если после этого граф не перестанет удовлетворять свойству. Проверять, соединены ли две вершины в графе, и поддерживать структуру при добавлении можно с помощью СНМ. Если в итоге граф будет состоять из двух компонент, то условие задачи выполнено, иначе, как мы показали, выполнить его невозможно. Асимптотика: $O(nm \cdot \alpha(nm))$.

Задача K. Жульничество при игре в камни

Пусть нам дано множество чисел A размера n, и мы хотим ответить на следующий вопрос: сколько существует подмножеств A размера k с суммой $s \leqslant S$? Эта стандартная задача на ДП, которая решается за $O(n^2 \cdot S)$: поочерёдно для каждого элемента x мы прибавляем $\mathrm{dp}[k][s]$ к $\mathrm{dp}[k+1][s+x]$, начиная с больших k.

Заметим следующий факт: поскольку итоговый массив dp не зависит от порядка, в котором мы рассматриваем элементы A, мы можем представить, что последним рассмотренным элементом был y, и, выполнив обратные действия, получить решение той же задачи для множества $A \setminus \{y\}$ за $O(n \cdot S)$.

Рассмотрим теперь исходную задачу. Пусть a и b — последовательности первого и второго игрока соответственно. Чтобы первый игрок выиграл после i ходов, необходимо, чтобы

$$S - b_i \leqslant (a_1 + \cdots + a_i) + (b_1 + \cdots + b_{i-1}) < S$$

Переберём $b_i =: x$. Пусть dp_a и dp_b — ответы на задачу, описанную выше, для a и $b \setminus \{x\}$. Зафиксируем i; пусть $S_a = a_1 + \cdots + a_i$, $S_b = b_1 + \cdots + b_{i-1}$.

Переберём S_b; мы хотим, чтобы $S - S_b - x \leqslant S_a < S - S_b$. Пусть cnt_a — это сумма $\mathrm{dp}_a[i][S_a]$ по всем таким S_a. cnt_a можно посчитать с помощью префсумм за $O(1)$.

С учётом всех зафиксированных величин количество подходящих перестановок равно

$$(\mathrm{cnt}_a \cdot i! \cdot (n-i)!) \cdot (\mathrm{dp}_b[i-1][S_b] \cdot (i-1)! \cdot (n-i)!)$$

Суммируя по всем x, i, S_b, получаем ответ. Асимптотика: $O(n^2 \cdot S)$.

Задача L. Зеркальное подмножество

Если точка симметрична относительно прямой, то она либо лежит на прямой, либо прямая является серединным перпендикуляром к этой и симметричной точкам. Поскольку ответ всегда (кроме $n = 1$) не меньше 2, искомая прямая либо является серединным перпендикуляром к двум точкам множества, либо проходит через две точки множества. Следовательно, нас интересует всего $O(n^2)$ прямых.

Чтобы сравнивать на равенство прямые, покажем, как приводить уравнение прямой к «каноническому» виду. Прямая задаётся уравнением $Ax + By + C = 0$, где A, B и C можно домножать на любую константу. Поделим A, B и C на их НОД и возьмём лексикографически минимальную из троек (A, B, C) и $(-A, -B, -C)$ как коэффициенты.

Все прямые, не являющиеся серединным перпендикуляром к каким-то двум точкам, можно перебрать отдельно. Для этого рассмотрим мультимножество из всех $n(n-1)/2$ прямых через две точки множества и возьмём из него самый часто встречающийся элемент.

Аналогично можно посчитать, для скольки пар точек некоторая прямая является серединным перпендикуляром. Остаётся посчитать для всех серединных перпендикуляров, сколько точек множества лежат непосредственно на них.

Переберём точку P. Если она лежит на серединном перпендикуляре к AB, то $|PA| = |PB|$. Переберём все точки $Q \neq P$ и сгруппируем их по расстоянию до P. Для всех пар точек внутри каждой группы P лежит на соответствующем серединном перпендикуляре.

Поскольку мы перебираем тройки точек, образующие равнобедренный треугольник, суммарно такое решение работает за $O(n^3)$. Можно заметить, однако, что на самом деле таких троек не может быть очень много. На практике удаётся построить пример с не более чем $O(n^2)$ таких троек. Отсюда получаем решение, работающее за $O(n^2 \log n)$.

Альтернативное решение использует метод вращающейся сканирующей прямой. У нас есть n точек и $O(n^2)$ прямых, для каждой из которых мы хотим узнать количество точек, лежащих на ней. Будем вращать воображаемую прямую, поддерживая порядок проекций точек на перпендикулярную ей прямую. Этот порядок меняется каждый раз, когда проекции двух точек меняются местами, то есть всего $O(n^2)$ раз. Когда воображаемая прямая параллельна одной из интересующих нас, достаточно сделать бинпоиск по проекциям на перпендикулярную прямую, чтобы ответить на запрос. Асимптотика этого решения: $O(n^2 \log n)$.

Задача M. Баян-баян-баян

Найдём кратчайший путь из 1 в n; он состоит из не более чем $n - 1$ рёбер.

Если мы удваиваем вес ребра, не лежащего на найденном пути, то длина кратчайшего пути из 1 в n не меняется. Таким образом, мы можем рассматривать только рёбра этого пути. Поочерёдно удвоим каждое из них и найдём максимальное получившееся расстояние от 1 до n. Разность между этим расстоянием и длиной исходного пути будет ответом.

Кратчайшие пути будем искать алгоритмом Дейкстры. Асимптотика: $O(nm \log m)$.