Задача A. Круглая карта

Пусть центр круга – точка O. Начальная точка A, точка первого удара – B. Пусть после нее пуля полетела в сторону точки C на окружности. Тогда из-за того, что угол падения равен углу отражения, отрезки BA и BC симметричны относительно прямой OB. А значит эти отрезки равны и $\angle AOB = \angle BOC$. Обозначим этот угол за φ. Получаем, что между ударами пуля поворачивается на угол φ относительно центра. Таким образом, после n ударов она повернётся на угол $n\varphi$. Тогда если координата точки A равна $(\cos(\psi), \sin(\psi))$, то итоговая координата будет равна $\cos(\psi + n\varphi), \sin(\psi + n\varphi)$.

Задача B. Очень большая игра

Рассмотрим какой-то тип бойцов. В нашей последовательности есть несколько отрезков подряд идущих бойцов этого типа. Заметим, что больше 2 отрезков за 1 операцию объединиться не может. Более того, любые 2 отрезка можно объединить (просто взять один их них и переместить к второму). То есть для каждого типа бойцов наибольший возможный отрезок в итоговой расстановке равен сумме двух самых длинных отрезков в исходной расстановке. То есть для каждого типа бойцов надо это посчитать и вывести максимум из полученных значений.

Для этого сначала за 1 проход по массиву разобьем его на отрезки подряд идущих солдат одного типа и сгруппируем по типам. Например, можно хранить $unordered_map$, в котором ключ – тип бойца, а значение – список отрезков. То есть за $O(n)$ мы получаем несколько списков с длинами отрезков. Причем суммарная длина этих списков равна $O(n)$.

Остается за $O(n)$ в каждом списке за линию найди 2 максимума и вывести наибольшую сумму. В итоге получаем решение за $O(n)$.

Задача C. Расстановка бойцов

Переформулируем задачу: мы хотим расположить на координатной прямой n отрезков с целочисленными координатами концов. Затем рассмотрим все точки, покрытые хотя бы одним отрезком. Для каждой такой точки посчитаем, сколько отрезков её покрывают. Получившееся мультимножество должно быть в точности $\{1, 2, \ldots, k\}$.

Т. к. должна существовать точка, покрытая k отрезками, то обязательно должно быть выполнено условие $n \geqslant k$.

Посчитаем сумму $1 + 2 + \ldots + k = \frac{k(k+1)}{2}$. Т. к. каждый отрезок вносит в эту сумму вклад хотя бы 1, то обязательно должно быть выполнено условие $n \leqslant \frac{k(k+1)}{2}$.

Таким образом, если $k > n$ или $n > \frac{k(k+1)}{2}$, то подходящего набора отрезков не существует.

Если $k \leqslant n \leqslant \frac{k(k+1)}{2}$, то ответ существует. Построим его следующим образом:

Возьмем следующий набор отрезков: $[1, 1], [1, 2], \ldots, [1, k]$. Очевидно, что условие на покрытие точек выполнено, и мы выбрали ровно k отрезков. Теперь будем выполнять следующий процесс: пока существует отрезок [a, b] такой, что $b > a$, заменим его на отрезки $[a, b-1]$ и $[b, b]$. Очевидно, что покрытие точек от этого не изменилось, и что каждый раз количество отрезков увеличивается на один, и в конце процесса оно будет равно $\frac{k(k+1)}{2}$. Значит, по свойству дискретной непрерывности, мы можем остановить этот процесс, когда отрезков будет ровно n.

Задача D. Сбор команды

Перефразируем условие: дано корневое дерево, в каждой вершине которого написано число. Также дано мультимножество A. Нужно посчитать количество вершин, мультимножество чисел в поддереве которых содержит множество A.

Обойдем дерево с помощью DFS. На момент выхода из вершины u мы хотим иметь посчитанным словарь, где ключами является множество чисел в поддереве u, а значениями — сколько раз это число встретилось в поддереве. Также помимо этого словаря мы хотим посчитать число $e[u]$ — количество ключей словаря, которых «уже достаточно». То есть посчитать количество чисел, которых в поддереве u содержится хотя бы столько же, сколько в мультимножестве A. Тогда ответ на задачу — количество вершин u таких, что $e[u]$ совпадает с количеством различных чисел в A.

Пусть при выполнении DFS мы обошли всех сыновей вершины u и посчитали для них словари и числа $e[v]$. Тогда для того, чтобы найти словарь для вершины u, нужно объединить словари всех

сыновей вершины u, а также добавить туда число $b[u]$. Сделаем это следующим образом: возьмем самый большой по размеру словарь среди сыновей и добавим в него значения из всех остальных словарей. При поочередном добавлении всех остальных значений в словарь будем пересчитывать значение $e[u]$.

Суммарно мы произведем $O(n \log n)$ операций со словарем. Доказательство этого факта — стандартная задача о переливании. Итоговая асимптотика будет $O(n \log n)$ или $O(n \log^2 n)$ в зависимости от используемой структуры данных.

Задача E. Оглуши меня полностью

Заметим что для реализации бесконечного контроля можно поступить тремя способами:

1. Сократить перезарядку c, так чтобы $c \leqslant a$.
2. Сократить перезарядку d, так чтобы $d \leqslant b$.
3. Сократить перезарядки d и c, так чтобы $c \leqslant a + b$ и $d \leqslant a + b$.

Исходя из ограничений в задаче, для каждого из вариантов воспользуемся задачей о рюкзаке с весами. Например в первом варианте посчитаем, можем ли мы набрать $\geqslant c - a$ времени сокращения перезарядки и минимальную стоимость этих артефактов.

Задача F. Прыгающая Пайпер

Заметим, что можно восстановить путь однозначно. Для этого нужно разбить задачу на два случая: пусть сейчас Пайпер в точке x', хочет попасть в x и текущий шаг это 2^i

- Если можно попасть в x, за 1 ход, то это очевидно нужно сделать, тк ищем минимальный путь
- Если нельзя, то точно будет шаг 2^{i+1}.

Заметим, что $2^{i+1} \equiv -2^{i+1} \pmod{2^{i+2}}$. Таким образом шаг 2^i последний определяющий остаток от деление суммарного изменения координаты по модулую 2^{i+2}. Следовательно, если $|x - x'| - 2^{i+1} \equiv 2^i \pmod{2^{i+2}}$, то нужно пытаться идти вправо, если $|x - x'| - 2^{i+1} \equiv -2^i \pmod{2^{i+2}}$, то влево, если ни то ни то, то очевидно прийти нельзя. Очевидно, что если не получится прыгнуть в желаемую сторону, то тоже прийти нельзя.
Таким образом можно проверить можно ли прийти в x' за $O(log(n))$ и того получим решение за $O(q \cdot log(n))$

Задача G. Лава в Джейлбрейке

Пусть есть n вулканов. Каждый i-й вулкан начинает извергаться в момент времени t_i из точки p_i с максимальной высотой лавы h_i. Лава распространяется со скоростью 1 влево и вправо по прямой; высота лавы в точке x в момент времени t для вулкана i: $\min\bigl(\max(0,\ t - t_i - |x - p_i|),\ h_i\bigr)$. Ответ в точке X в момент времени T — максимум по всем вулканам.

Для каждого запроса (T, X) вклад вулкана i равен нулю, если $T < t_i$ или $|X - p_i| > T - t_i$. В противном случае вклад равен $v_i = \min((T - t_i) - |X - p_i|,\ h_i)$.

Если расписать $(T - t_i) - |X - p_i|$, то если $X \geqslant p_i$, получаем $(T - X) - (t_i - p_i)$. Если $X < p_i$, то $(T + X) - (t_i + p_i)$. Пусть $A_i = t_i - p_i$, $B_i = t_i + p_i$. Для каждого запроса заведём $u_1 = T - X$, $u_2 = T + X$.

Получаем две независимые задачи. В первой надо для каждого u_1 найти максимум среди вулканов, у которых $A_i \leqslant u_1$, по выражению $\min(u_1 - A_i, h_i)$. Аналогично для u_2 и B_i.

Для этого удобно обработать все события offline. Для первой части отсортируем все A_i и u_1, сожмём координаты. Будем идти по возрастанию u_1 и поддерживать в дереве отрезков для всех $A_i \leqslant u_1$ два значения:

- максимум h_i (то есть максимум по плато),
- максимум A_i (для фронта, где $u_1 - A_i < h_i$).

Тогда ответ на запрос — максимум из $\max(h_i)$ по $A_i \leqslant u_1$ и $u_1 - \min(A_i)$ по $A_i \leqslant u_1$ (учитывая ограничение на высоту h_i). То же делаем во второй части по B_i и u_2.

Таким образом, задача сводится к двум offline-проходам с поддержанием максимумов по отрезкам в дереве отрезков. В начале делаем сжатие координат по A_i и B_i (размер не больше n). Все события сортируются по u, по мере продвижения добавляются соответствующие значения в дерево отрезков. Ответы на запросы — максимум по нужному префиксу.

Полная сложность $O((n+q)\log n)$. Итоговый ответ для каждого запроса — максимум из двух направлений.

Задача H. Трофейная дорога Эдгара

В задаче карточки можно представить как рёбра в графе, состоящем из 6 вершин. Тогда необходимо было найти самый длинный путь по рёбрам, не проходя по одному ребру дважды. В таком графе всего $\frac{6\cdot 5}{2} = 15$ рёбер, которые можно перебрать, удаляя по одному. Итоговый путь — это эйлеров путь в графе. Тогда можно перебрать все рёбра, по которым мы не пойдём, и проверить компоненты связности и чётность степени вершин. Сложность — $O(2^{15})$.

Также можно заметить, что если у нас есть k вершин с нечётной степенью в одной компоненте, то ответ не может быть больше $m - \frac{k-2}{2}$ рёбер, где m — количество ребер в компоненте. На самом деле, в данных ограничениях такой максимум достигается. Доказательство этого утверждения остаётся читателю в качестве упражнения.

Задача I. Безопасный прыжок

Заметим, что по 3 точкам, через которые проходит очередная траектория: $(x_i, y_i), (x_{i+1}, y_{i+1}), (x'_i, y'_i)$. Заметим, что по ним однозначно восстанавливается парабола, а именно квадратный трехчлен($a \cdot x^2 + b \cdot x + c = 0$), задающий ее. Для этого нужно решить систему линейных уравнений:

- $a \cdot x_i^2 + b \cdot x_i + c = y_i$
- $a \cdot x_{i+1}^2 + b \cdot x_{i+1} + c = y_{i+1}$
- $a \cdot (x'_i)^2 + b \cdot x'_i + c = y'_i$

Далее, имея трехчлен, можно найти самую низкую точку траектории. Для этого нужно воспользоваться формулой для оптимума парабола ($\frac{-b}{2a}$). Если он лежит в отрезе между двумя платформами для прыжка, то нужно взять минимум из него и крайних точек (на случай если ветви параболы направлены вверх). Если же нет, то хватит просто крайних точек, что становится очевидно, если посмотреть на график параболы. Взяв минимум по всем таким минимумам, получим ответ. И того получили решение за $O(n)$

Задача J. Зеркальная Галерея

В задаче можно заметить, что, чтобы пройти 1 метр нужно сделать 1 отражение, поэтому достаточно, вывести изначальное число n.