**Задачи на отрезки на прямой**

Рассмотрим следующую задачу. На прямой дано некоторое количество отрезков. Необходимо найти, какое наибольшее количество отрезков может одновременно пересекаться в одной точке.

Для решения этой задачи достаточно понять, что ответом (точкой, в которой пересекается наибольшее количество отрезков) является один из концов отрезков (иначе можно «передвинуть» ответ так, чтобы он совпал с каким либо из концов отрезков). Но перебирая все концы отрезков и для каждого из них подсчитывая количество пересечений с другими отрезками (то есть перебирая все другие отрезки), получится решения сложности O(n2), где n — количество данных нам отрезков. Можно решить задачу за меньшую сложность.

Для этого будем двигаться по числовой прямой слева направо, считая количество пересечений отрезков с текущим положением рассматриваемой точки. Эта величина (количество пересечений) будет увеличиваться на 1, при прохождении через начало отрезка, и уменьшаться на 1, при прохождении через конец отрезка. Только эти точки представляют интерес, поскольку в этих точках меняется значение интересующего нас показателя. Будем называть интересующие нас точки **событиями**, то есть событиями являются начала и концы отрезков.

В этом случае для решения задачи нам нужно считать координаты начал и концов всех отрезков и составить список событий на прямой. Затем необходимо упорядочить список событий по возрастанию их координаты, пройти по списку событий, увеличивая значение счетчика на 1, при встрече начала отрезка, и уменьшая значение счетчика на 1 на концах отрезков. Одновременно с этим нужно искать максимальное значение счетчика, что и будет ответом на нашу задачу.

Переменная типа «событие» должна хранить в себе минимум два значения: координату точки x и тип события, которое позволяет отличать события типа «начало отрезка» от события типа «конец отрезка». Для представления событий будем использоваться кортежи, при этом поскольку нам необходимо сортировать события по возрастанию координаты, а кортежи сортируются в лексикографическом порядке (то есть прежде всего по первому элементу кортежа, а при равном значении первого элемента — по второму), то первым элементом кортежа будет координата x события.

При равном значении координаты x необходимо, чтобы сначала обрабатывались начала отрезков, а потом концы отрезков, в этом случае если в одной точке есть и начала отрезков, и концы отрезков, то мы посчитаем эту точку, как принадлежащую всем отрезкам, и начинающимся, и заканчивающимся в этой точке. Поэтому будем кодировать начала отрезков числом -1, а концы отрезков числом 1, это будет вторым элементом кортежа типа событие.

Итак, считывание всех концов отрезков и создание списка событий можно реализовать следующим образом:

**Пример реализации на языке C++**

**int n;**

**cin >> n;**

**vector <pair <int, int> > Events;**

**for (int i = 0; i < n; ++i) {**

**int l, r;**

**cin >> l >> r;**

**Events.push\_back(make\_pair(l, -1));**

**Events.push\_back(make\_pair(r, 1));**

**}**

**Пример реализации на языке Python**

**n = int(input())**

**Events = []**

**for i in range(n):**

**l, r = map(int, input().split())**

**Events.append((l, -1))**

**Events.append((r, 1))**

В некоторых задачах может оказаться необходимым сохранение информации о номере отрезка, в этом случае в кортеж добавляется дополнительное поле, в которое необходимо записать значение переменной i.

Дальнейшее решение задачи просто. Заведем переменную count для подсчета количества отрезков, которые пересекаются. Тогда для поддержания значения count необходимо из значения count вычитать значение типа отрезка (которое равно -1 для начала и 1 для конца) и искать максимальное значение величины count.

**Пример реализации на языке С++**

**sort(Events.begin(), Events.end());**

**int count = 0;**

**int ans = 0;**

**for (auto event in Events) {**

**int x = event.first;**

**int type = events.second;**

**count -= type;**

**ans = max(ans, count);**

**}**

**Пример реализации на языке Python**

**count = 0**

**ans = 0**

**for x, type in sorted(Events):**

**count -= type**

**ans = max(ans, count)**

Слегка модифицировав это решение, можно научиться, например, считать длину объединения всех отрезков. Объединение отрезков представляет собой также набор отрезков, концы которых совпадают с какими-либо событиями (то есть с концами данных отрезков). Тогда началом очередного отрезка из объединения данных отрезков будет точка, в которой значение count меняется с 0 на 1, что можно также проверить условием count равно 1 и тип события — начало отрезка (после обработки отрезка), а конец объединения отрезка — точка, в которой значение count становится равно 0. В случае, если встретилась точка, являющаяся началом одного из отрезков объединения, запомним ее координату в переменной last, а если встретилась точка — конец одного из отрезков объединения, вычтем из ее координат last и добавим эту величину к ответу.

**Пример реализации на языке C++**

**sort(Events.begin(), Events.end());**

**int count = 0;**

**int ans = 0;**

**int last;**

**for (auto event in Events) {**

**int x = event.first;**

**int type = events.second;**

**count -= type;**

**if (count == 1 && type == -1)**

**last = x;**

**if (count == 0)**

**ans += x — last;**

**}**

**Пример реализации на языке Python**

**count = 0**

**ans = 0**

**for x, type in sorted(Events):**

**count -= type**

**if count == 1 and type == -1:**

**last = x**

**if count == 0:**

**ans += x — last**