Московский Авиационный Институт

(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

**Курсовой проект по предмету**

**«Дискретный анализ»**

# **Персистентные структуры данных**

Студент: Куценко Максим Дмитриевич

Группа: М8О–312Б-22

Преподаватель: Сорокин С.А.

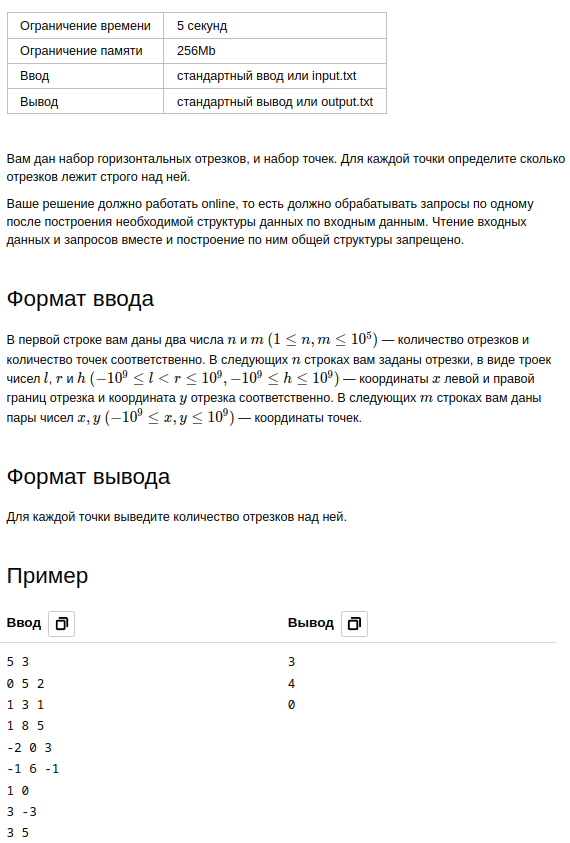
Оценка: \_\_\_\_\_\_\_\_\_\_

Дата: \_\_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_

Москва, 2024.

**Условие**

****

**Метод решения**

Для данной задачи нам необходимо воспользоваться персистентностью — сохранением всех изменений в качестве отдельных версий с дальнейшей возможностью доступа к ним. В качестве подходящей структуры выберем персистентное дерево отрезков с суммой в внутренних вершинах.

Построение дерева будем осуществлять следующим образом: считываем все отрезки и переводим их в «события», ставя для левой границы отрезка по x величину 1, а для координаты+1 по x правой границы величину -1. Сортируем данные события по иксу. Берём уникальные координаты по y, сортируем их, сжимаем эти координаты. Берём из событий отсортированные координаты по x.

Создаём дерево в зависимости от числа уникальных координат y и для каждого события создаём свою версию, осуществляя изменение по соответствующему индексу в дереве, нулевая версия — пустая. Индекс выбирается из сжатия вершины.

Далее, для каждой точки из запроса: для координаты y находим первую координату, который больше чем она, не нашли - тогда 0 отрезков над ней; переводим найденную координату в сжатые. Для координаты x находим последний элемент из событий, который не больше чем x. Координата меньше всех или больше всех - тогда 0 отрезков над точкой. И в конце делаем запрос с позицией по x в качестве версии и сжатой координатой по y в качестве искомого элемента.

**Описание программы**

Дерево хранит внутри себя вектор векторов cords, где для каждого элемента из внешнего вектора (вершины дерева по сжатой координате) хранится вектор версий versions. Каждая версия хранит в себе: текущее значение вершины val, версию левого ребёнка, версию правого ребёнка. Корень находится в cords[1]; в cords[2\*i] и cords[2\*i+1] хранятся соответственно левый и правый дети i-й вершины. Размер cords при создании структуры задаём как 4\*n, где n — число уникальных координат по y.

При каждом изменении дерева через update создаётся новый корень и вызывается рекурсивная функция update\_internal, где при прохождении до изменяемого листа создаются новые версии детей текущей вершины и меняются соответствующие left и right.

При просмотре дерева через query аналогично вызываем рекурсивную функцию query\_internal, где проходимся по дереву, ориентируясь на версии детей left и right из выбираемой версии текущей вершины.

Сжатие координат производим через создание и использование std::unordered\_map, которую заполняем по отсортированному вектору уникальных вершин отрезков y.

При запросе производим поиск по векторам для выбора версии и индекса через std::upper\_bound.

**Дневник отладки**

1) Сделал обработку событий, но все события просто закинул в многомерный unordered\_map, по которому проходился при каждом запросе. Не прошло по времени.

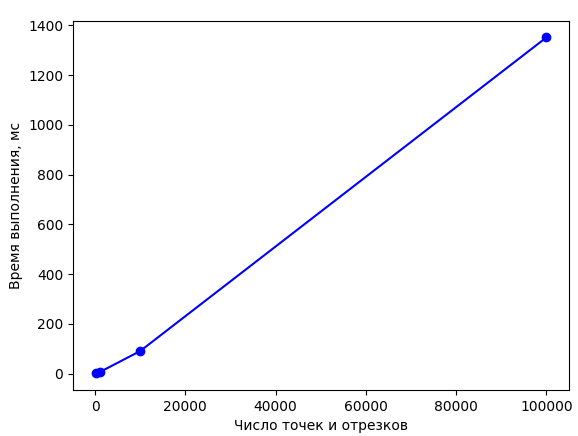
2) Сделал персистентный массив — персистентность по y, листы по x. Создавал новые версии только при переходе к новому y. Не прошло по времени и памяти.

3) Сделал персистентное дерево отрезков — персистентность по x, листы по y. Создавал новые версии только при переходе к новому x. Не прошло по времени.

4) Убрал проверку на переход к новому x, перейдя к созданию новых версий при каждом изменении.

**Тест производительности**

График времени выполнения

Алгоритмическая сложность данного алгоритма — O( NlogN + MlogN ), сложность по памяти — O(NlogN), где N — число отрезков, M — число вершин.

**Выводы**

Ознакомился с персистентностью и различными персистентными структурами данных. Изучил, что из себя представляет дерево отрезков и его персистентная версия, просмотрел различные его реализации — на массиве/векторе или указателях, ленивую версию дерева отрезков. Вспомнил метод сканирующей прямой.

Применил данные знания на практике для решения данного курсового проекта.