Содержание

Пример работы с файлами.

Если вы не умеете читать/выводить данные, или открывать файлы, воспользуйтесь примерами. http://acm.math.spbu.ru/~sk1/algo/sum/

Пример работы с файлами.

В некоторых задачах большой ввод и вывод. Про ввод-вывод в C++:
http://acm.math.spbu.ru/~sk1/algo/input-output/cpp_common.html
Имеет смысл пользоваться супер быстрым вводом-выводом. Две версии:
http://acm.math.spbu.ru/~sk1/algo/input-output/io_export.cpp.html
http://acm.math.spbu.ru/~sk1/algo/input-output/fread_write_export.cpp.html

Выделение памяти.

В некоторых задачах нужен STL, который активно использует динамическую память (set-ы, map-ы) переопределение стандартного аллокатора ускорит вашу программу:
http://acm.math.spbu.ru/~sk1/algo/memory.cpp.html

Обязательные задачи

В этих задачах нужно использовать дерево отрезков каких-либо структур данных.
В первой нельзя искользовать сканлайн!

Задача 19A. Перестановки [1 sec, 256 mb]

Вася выписал на доске в каком-то порядке все числа от 1 по N, каждое число ровно по одному разу. Количество чисел оказалось довольно большим, поэтому Вася не может окинуть взглядом все числа. Однако ему надо всё-таки представлять эту последовательность, поэтому он написал программу, которая отвечает на вопрос — сколько среди чисел, стоящих на позициях с x по y, по величине лежат в интервале от k до l. Сделайте то же самое.

Формат входных данных

В первой строке лежит два натуральных числа — $1 \leqslant N \leqslant 100\,000$ — количество чисел, которые выписал Вася и $1 \leqslant M \leqslant 100\,000$ — количество вопросов, которые Вася хочет задать программе. Во второй строке дано N чисел — последовательность чисел, выписанных Васей. Далее в M строках находятся описания вопросов. Каждая строка содержит четыре целых числа $1 \leqslant x \leqslant y \leqslant N$ и $1 \leqslant k \leqslant l \leqslant N$.

Формат выходных данных

Выведите M строк, каждая должна содержать единственное число — ответ на Васин вопрос.

Пример

permutation.in	permutation.out
4 2 1 2 3 4 1 2 2 3 1 3 1 3	1 3

Задача 19B. Count Offline [2 sec, 256 mb]

Вам дано множество точек на плоскости.

Нужно уметь отвечать на два типа запросов:

- `+ x y` — добавить в множество точку (x, y).
- `?` x_1 y_1 x_2 y_2 — сказать, сколько точек лежит в прямоугольнике $[x_1..x_2] \times [y_1..y_2]$. Точки на границе и в углах тоже считаются. $x_1 \leqslant x_2$, $y_1 \leqslant y_2$.

Формат входных данных

Число точек N ($1 \leqslant N \leqslant 50\,000$). Далее N точек. Число запросов Q ($1 \leqslant Q \leqslant 100\,000$). Далее Q запросов. Все координаты от 0 до 10^9.

Формат выходных данных

Для каждого запроса `GET` одно целое число — количество точек внутри прямоугольника.

Пример

countoffline.in	countoffline.out
4 0 0 1 0 0 1 1 1 5 ? 0 1 1 2 + 1 2 + 2 2 ? 1 0 2 2 ? 0 0 0 0	2 4 1

Бонус

Нужно знать дерево Фенвика

Задача 19C. Points on the plane [0.5 sec, 256 mb]

Есть квадратная клетчатая плоскость состоящая из $n \times n$ клеток ($1 \leqslant n \leqslant 1000$). Изначально в каждой клетке записано значение ноль. Ваша задача — написать программу, умеющую отвечать на следующие запросы:

- ADD x y — увеличить значение в ячейке x, y на 1.
- GET x_1 y_1 x_2 y_2 — вернуть сумму значений в прямоугольнике с углами в x_1, y_1 и x_2, y_2 соответственно.

Формат входных данных

В первой строке входного файла содержится два числа — n и k — размер доски и число запросов соответственно. Следующие k строк содержат сами запросы. Гарантируется, что общее число запросов не превосходит 300 000.

Формат выходных данных

Для каждого запроса типа GET выведите в отдельную строку одно целое число — ответ на соответствующий запрос.

Примеры

fenwick.in	fenwick.out
5 15 ADD 1 1 ADD 2 2 ADD 3 3 ADD 4 4 ADD 5 5 ADD 1 5 ADD 2 4 ADD 3 3 ADD 4 2 ADD 5 1 GET 1 1 5 5 GET 2 1 5 5 GET 1 2 5 5 GET 2 2 4 4 GET 3 3 3 3	10 8 8 6 2

Задача 19D. Фыр-Фыр-Фенвик [2 sec, 256 mb]

Даны n точек с весами на плоскости. Каждая задаётся тремя числами x_i, y_i, w_i (координаты и вес). Вам нужно обработать m запросов двух типов:

- `get rx ry` – посчитать сумму весов точек, у которых $x_i \leqslant rx$ и $y_i \leqslant ry$.
- `change i z` – задать i-й точке новый вес равный z.

Формат входных данных

На первой строке число n ($1 \leqslant n \leqslant 100\,000$). На следующих n строках тройки целых чисел x_i, y_i, w_i ($0 \leqslant x_i, y_i, w_i < 10^9$). Следующая строка содержит количество запросов m ($1 \leqslant m \leqslant 300\,000$). На следующих m строках описания запросов в формате `get rx, ry` и `change i z`. Здесь $1 \leqslant i \leqslant n$, а остальные числа целые от 0 до 10^9-1.

Формат выходных данных

Для каждого запроса типа "`get`" выведите одно целое число на отдельной строке — ответ на запрос.

Пример

fffenwick.in	fffenwick.out
3 1 1 10 2 3 100 3 2 1000 4 get 2 3 change 2 200 get 3 3 get 3 2	110 1210 1010