Задача A. Художник

Имя входного файла:	`painter.in`
Имя выходного файла:	`painter.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Не успев дорисовать свой гениальный футуристический шедевр, М. Калевич увлёкся рисованием одномерных чёрно-белых картин. Он пытается найти оптимальные местоположение и количество чёрных участков картины. Для этого он проводит на прямой белые и чёрные отрезки и после каждой из таких операций хочет знать количество чёрных отрезков на получившейся картине и их суммарную длину.

Изначально прямая белая. Ваша задача — написать программу, которая после каждой такой операции выводит в выходной файл интересующие художника данные.

Формат входных данных

В первой строке входного файла содержится общее количество нарисованных отрезков ($1 \leqslant N \leqslant 100\,000$). В последующих N строках содержится описание операций. Каждая операция описывается строкой вида $c\ x\ l$, где c — цвет отрезка (‘`W`’ для белых отрезков и ‘`B`’ для чёрных), а сам отрезок имеет вид $[x; x + l]$, причём координаты обоих концов — целые числа, по модулю не превосходящие 500 000. Длина задаётся положительным целым числом.

Формат выходных данных

После выполнения каждой из операций необходимо вывести в выходной файл на отдельной строке количество чёрных отрезков на картине и их суммарную длину, разделённые одним пробелом.

Примеры

painter.in	painter.out
7 W 2 3 B 2 2 B 4 2 B 3 2 B 7 2 W 3 1 W 0 10	0 0 1 2 1 4 1 4 2 6 3 5 0 0

Задача B. Мега-инверсии

Имя входного файла:	`mega.in`
Имя выходного файла:	`mega.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	64 мегабайта

Инверсией в перестановке $p_1, p_2, \ldots, p_N$ называется пара (i, j) такая, что $i < j$ и $p_i > p_j$. Назовём *мега-инверсией* в перестановке $p_1, p_2, \ldots, p_N$ тройку (i, j, k) такую, что $i < j < k$ и $p_i > p_j > p_k$. Напишите алгоритм для быстрого подсчёта количества мега-инверсий в перестановке.

Формат входных данных

Первая строка входного файла содержит целое число N ($1 \leqslant N \leqslant 100\,000$). Следующие N чисел описывают перестановку: $p_1, p_2, \ldots, p_N$ ($1 \leqslant p_i \leqslant N$), все p_i попарно различны. Числа разделяются переводами строк.

Формат выходных данных

Единственная строка выходного файла должна содержать одно число, равное количеству мега-инверсий в перестановке $p_1, p_2, \ldots, p_N$.

Примеры

mega.in	mega.out
4 4 3 2 1	4

Задача C. Без сказок

Имя входного файла:	`minsumseg.in`
Имя выходного файла:	`minsumseg.out`
Ограничение по времени:	5 секунд
Ограничение по памяти:	64 мегабайта

У этой задачи нет легенды. Вам дана последовательность из N целых чисел и M запросов одного из двух типов:

- *change* ps, val — заменить число стоящее на позиции ps числом val.
- *get* l, r — найти подотрезок отрезка $[l, r]$ с максимальной суммой.

Обратите внимание на факт, что по определению, пустой отрезок является подотрезком любого отрезка.

Формат входных данных

Первая строка содержит два целых положительных числа N и M не превосходящих $300\,000$. Следующая строка содержит N целых чисел - изначальную последовательность. Следующие M строк содержат запросы в формате описанном в условии. Гарантируется, что все запросы корректны и все значения в последовательности в любой момент не превосходят по модулю 10^9. Используется индексация от 1.

Формат выходных данных

Для каждого запроса *get* выведите одно число — сумму чисел на подотрезке являющемся ответом на данный запрос.

Примеры

minsumseg.in	minsumseg.out
4 2 -5 2 -1 2 get 1 2 get 1 4	2 3

Задача D. Перестановки strike back

Имя входного файла:	`permutation2.in`
Имя выходного файла:	`permutation2.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Вася выписал на доске в каком-то порядке все числа от 1 по N, каждое число ровно по одному разу. Иногда он стирает какое-то число и записывает на его место другое. Количество чисел, выписанных Васей, оказалось довольно болшим, поэтому Вася не может окинуть взглядом все числа. Однако ему надо всё-таки представлять эту последовательность, поэтому он написал программу, которая в любой момент 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 $1 \leqslant M \leqslant 100\,000$ — суммарное количесто вопросов и изменений сделанных Васей. Во второй строке дано N чисел — последовательность чисел, выписанных Васей. Далее в M строках находятся описания вопросов. Каждый запрос на изменение числа в некоторой позиции начинается со слова `SET` и имеет вид `SET a b` ($1 \leqslant a \leqslant N$, $1 \leqslant b \leqslant N$). Это означает, что Вася изменил число, записанное в позиции a на число b. Каждый Васин вопрос начинается со слова `GET` и имеет вид `GET x y k l` ($1 \leqslant x \leqslant y \leqslant N$, $1 \leqslant k \leqslant l \leqslant N$).

Формат выходных данных

Для каждого Васиного вопроса выведите единственное число — ответ на Васин вопрос.

Примеры

permutation2.in	permutation2.out
4 4 1 2 3 4 GET 1 2 2 3 GET 1 3 1 3 SET 1 4 GET 1 3 1 3	1 3 2

Задача E. Фенечка

Имя входного файла:	`bracelet.in`
Имя выходного файла:	`bracelet.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Саша находится в процессе творческого поиска. Она хочет сплести ещё одну фенечку, но испытывает сложности при выборе цветов. Сейчас все n ниток, которые она планирует использовать для плетения, выложены в ряд. В процессе размышления Саша время от времени заменяет нитку одного цвета ниткой другого, а также для проверки того, что узор получается тем, который подразумевается, проверяет, что некоторые последовательности цветов ниток равны.

Напишите программу, которая автоматизирует эти проверки.

Формат входных данных

В первой строке входного файла записаны два целых числа n и k — количество ниток в фенечке и запросов к программе, соответственно. $1 \leqslant n, k \leqslant 100\,000$. Во второй строке записана строка из n символов — цвета ниток в начальном состоянии. Каждый цвет обозначается строчной или прописной буквой латинского алфавита или цифрой.

В следующих k строках заданы запросы двух видов:

- «`*` i c» — заменить нитку с номером i на нитку цвета c,
- «`?` i j len» — проверить, равны ли последовательности цветов ниток, начинающиеся в позициях i и j и имеющие длину len.

Формат выходных данных

Для каждого запроса второго вида выведите «`+`», если последовательности равны, или «`-`» в противном случае.

Примеры

bracelet.in	bracelet.out
7 4 abacaba ? 1 5 3 * 6 c ? 2 6 2 ? 3 5 3	+-+
10 5 vznwsempqf * 8 c * 2 z ? 5 4 5 * 6 p ? 5 4 1	--

Задача F. Ферма

Имя входного файла:	`segtree2d.in`
Имя выходного файла:	`segtree2d.out`
Ограничение по времени:	10 секунды
Ограничение по памяти:	256 мегабайт

Настала весна и фермер решил заняться удобрением своего земельного участка размерами $x \times y$ метров. Для этого он закупил удобрения. До начала посевов остаётся n дней, и фермер хочет успеть сделать как можно больше.

За день фермер может одну из следующих вещей:

- увеличить продуктивность прямоугольного участка земли со сторонами, параллельными осям координат с углами (x_1, y_1) и (x_2, y_2) на значение w
- посчитать суммарную продуктивность участка $(x_1, y_1) — (x_2, y_2)$

Удобрять фермер любит сам, а вот заниматься скучными расчетами ему не интересно. Помогите ему в этом.

Формат входных данных

В первой строке входного файла записаны числа x и y ($1 \leqslant x, y \leqslant 1000$). В следующей строке написано количество оставшихся до начала посевов дней n ($1 \leqslant n \leqslant 100000$). Следующие n строк описывают действия фермера в соответственный день в следующем формате:

- 1 x_1 y_1 x_2 y_2 w — фермер удобряет участок. ($1 \leqslant x_1 \leqslant x_2 \leqslant x$, $1 \leqslant y_1 \leqslant y_2 \leqslant y, -10000 \leqslant w \leqslant 10000$)
- 2 x_1 y_1 x_2 y_2 — фермер просит посчитать плодородность участка. ($1 \leqslant x_1 \leqslant x_2 \leqslant x$, $1 \leqslant y_1 \leqslant y_2 \leqslant y$)

Формат выходных данных

Для каждого запроса плодородности участка в отдельной строке выведите плодородность этого участка.

Примеры

segtree2d.in	segtree2d.out
8 8 3 1 2 2 8 8 2 1 1 1 2 2 1 2 2 2 2 2	3

Задача G. K-ый максимум

Имя входного файла:	`kthmax.in`
Имя выходного файла:	`kthmax.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Напишите программу, реализующую структуру данных, позволяющую добавлять и удалять элементы, а также находить k-й максимум.

Формат входных данных

Первая строка входного файла содержит натуральное число n — количество команд ($n \leqslant 100\,000$). Последующие n строк содержат по одной команде каждая. Команда записывается в виде двух чисел c_i и k_i — тип и аргумент команды соответственно ($|k_i| \leqslant 10^9$). Поддерживаемые команды:

- $+1$ (или просто 1): Добавить элемент с ключом k_i.
- 0: Найти и вывести k_i-й максимум.
- -1: Удалить элемент с ключом k_i.

Гарантируется, что в процессе работы в структуре не требуется хранить элементы с равными ключами или удалять несуществующие элементы. Также гарантируется, что при запросе k_i-го максимума, он существует.

Формат выходных данных

Для каждой команды нулевого типа в выходной файл должна быть выведена строка, содержащая единственное число — k_i-й максимум.

Примеры

kthmax.in	kthmax.out
11 +1 5 +1 3 +1 7 0 1 0 2 0 3 -1 5 +1 10 0 1 0 2 0 3	7 5 3 10 7 3