Содержание

Обратите внимание, входные данные лежат в **стандартном потоке ввода** (он же stdin), вывести ответ нужно в **стандартный поток вывода** (он же stdout).

Обратите внимание на `GNU C++` компиляторы с суффиксом `inc`.
Подни можно пользоваться дополнительной библиотекой (`optimization.h`).
То есть, использовать быстрый ввод-вывод: **пример про числа и строки**.
И быструю аллокацию памяти (ускоряет vector-set-map-весь-STL): пример.

Для тех, кто хочет разобраться, как всё это работает.
Короткая версия быстрого ввода-вывода (тык) и короткая версия аллокатора (тык).

Must have

Задача 5A. Пересортировка подсчётом [0.2, 256]

У вас есть изначально пустой массив. Вам нужно несколько раз обработать запрос вида вставить в массив целое число от 1 до 100 так, чтобы массив остался отсортированным и вывести, на какую позицию был добавлен элемент.

Если возможных хороших позиций для вставки несколько, выбирайте самую правую.

Формат входных данных

В первой строке целое число n ($1 \leqslant n \leqslant 10^5$) — количество запросов.

Следующая строка содержат сами запросы — n целых чисел от 1 до 100.

Формат выходных данных

Для каждого запроса выведите позицию вставки. Позиция вставки в начало массива равна 0, позиция вставки в конец массива длины n равна n.

Примеры

stdin	stdout
9 2 2 2 3 2 3 4 3 1	0 1 2 3 3 5 6 6 0

Замечание

Пояснение к первому примеру:

Жирным выделы только что вставленные в массив элементы.

$[\mathbf{2}]$
$[2, \mathbf{2}]$
$[2, 2, \mathbf{2}]$
$[2, 2, 2, \mathbf{3}]$
$[2, 2, 2, \mathbf{2}, 3]$
$[2, 2, 2, 2, 3, \mathbf{3}]$
$[2, 2, 2, 2, 3, 3, \mathbf{4}]$
$[2, 2, 2, 2, 3, 3, \mathbf{3}, 4]$
$[\mathbf{1}, 2, 2, 2, 2, 3, 3, 3, 4]$

Подсказка по решению

Мы знаем сортировку подсчётом. Эту задачу проще всего решать как раз подсчётом.

Задача 5B. Словарная Задача [0.3, 256]

Нужно обрабатывать запросы двух типов: добавить число x в мультимножество, удалить число x из мультимножества.

Формат входных данных

Количество запросов n ($1 \leqslant n \leqslant 100\,000$).

Следующие n строк содержат запросы двух типов «`+` x» и «`-` x».

Все числа целые от 1 до 10^9.

Формат выходных данных

Если после очередного запроса «`-` x» чисел x в мультимножестве больше не осталось, выведите «`after query` i `number` x `disappeared`», где i — номер запроса. Если при очередном запросе «`-` x» числа x уже нет, выведите «`query #`i`: can not delete` x».

Примеры

stdin	stdout
8 - 7 + 7 + 7 - 7 - 7 + 2 - 2 - 2	query #1: can not delete 7 after query 5 number 7 disappeared after query 7 number 2 disappeared query #8: can not delete 2

Задачи здорового человека

Задача 5C. Мультиграф [0.4, 256]

Дан неориентированный невзвешенный граф. В графе возможны петли и кратные рёбра. Постройте такой новый граф без петель и кратных рёбер, что для любых двух вершин в нём расстояние равно расстоянию в исходном графе. Если вершины не связны, расстояние между ними бесконечность.

Формат входных данных

На первой строке число вершин n и число рёбер m ($1 \leqslant n, m \leqslant 100\,000$). Следующие m строк содержат пары чисел от 1 до n – рёбра графа.

Формат выходных данных

Новый граф в таком же формате. Рёбра можно выводить в произвольном формате.

Примеры

stdin	stdout
3 5 1 1 1 3 2 1 1 2 2 3	3 3 1 2 2 3 3 1

Задача 5D. Обмен [1, 256]

Пусть все натуральные числа исходно организованы в список в естественном порядке. Разрешается выполнить следующую операцию: $swap(a, b)$. Эта операция возвращает в качестве результата расстояние в текущем списке между числами a и b и меняет их местами.

Задана последовательность операций *swap*. Требуется вывести в выходной файл результат всех этих операций.

Формат входных данных

Первая строка входного файла содержит число n ($1 \leqslant n \leqslant 200\,000$) — количество операций. Каждая из следующих n строк содержит по два числа в диапазоне от 1 до 10^9 — аргументы операций *swap*.

Формат выходных данных

Для каждой операции во входном файле выведите ее результат.

Пример

stdin	stdout
4 1 4 1 3 4 5 1 4	3 1 4 2

Замечание

C++:`unordered_map`, python:`dict()`

На C++ можно потренироваться и написать свою.

Задача 5E. Обмен [0.16, 256]

Такая же задача, как предыдущая, но с жёстким TL.

Подсказка по решению

Придётся писать свою хеш-таблицу.
Самая быстрая и простая хеш-таблица – хеш-таблица с открытой адресацией.

Вам нужно ровно 2 обращение к хеш-таблице на запрос. Самое важное – функция `getIndex(x)` (см. конспект ПМИ). Вы знаете, какая хеш-функция подойдёт, какая не подойдёт (см. конспект ПМИ).

Для искателей острых ощущений

Задача 5F. Ближайшая точка на прямой на плоскости [0.5, 256]

Вам дан массив точек на плоскости a. Каждая точка описывается парой целых чисел. Гарантируется, что все точки, заданные элементами массива a лежат на одной прямой. Вам необходимо ответить на q запросов вида «какая из точек в массиве находится ближе всего к точке (x_i, y_i)». Если подходящих точек несколько, выведите наименьшую в лексикографическом порядке (сравните по координате x, в случае равенства, по координате y).

Формат входных данных

На первой строке длина массива n ($1 \leqslant n \leqslant 10^5$). На следующих n строках по два целых числа a_i, b_i, описывающие точки ($-10^5 \leqslant a_i, b_i \leqslant 10^5$). На третьей строке количество запросов q ($1 \leqslant q \leqslant 10^4$). На следующих q строках по два целых числа x_i, y_i, которые задают i-й запрос ($-10^5 \leqslant x_i, y_i \leqslant 10^5$).

Формат выходных данных

Выведите q строк, в i-й строке выведите точку из массива a, которая находится ближе всех к точке (x_i, y_i).

Примеры

stdin	stdout
5 3 2 -1 10 2 4 5 -2 5 -2 4 -3 7 6 5 5 -3 2 -1	-1 10 2 4 5 -2 3 2

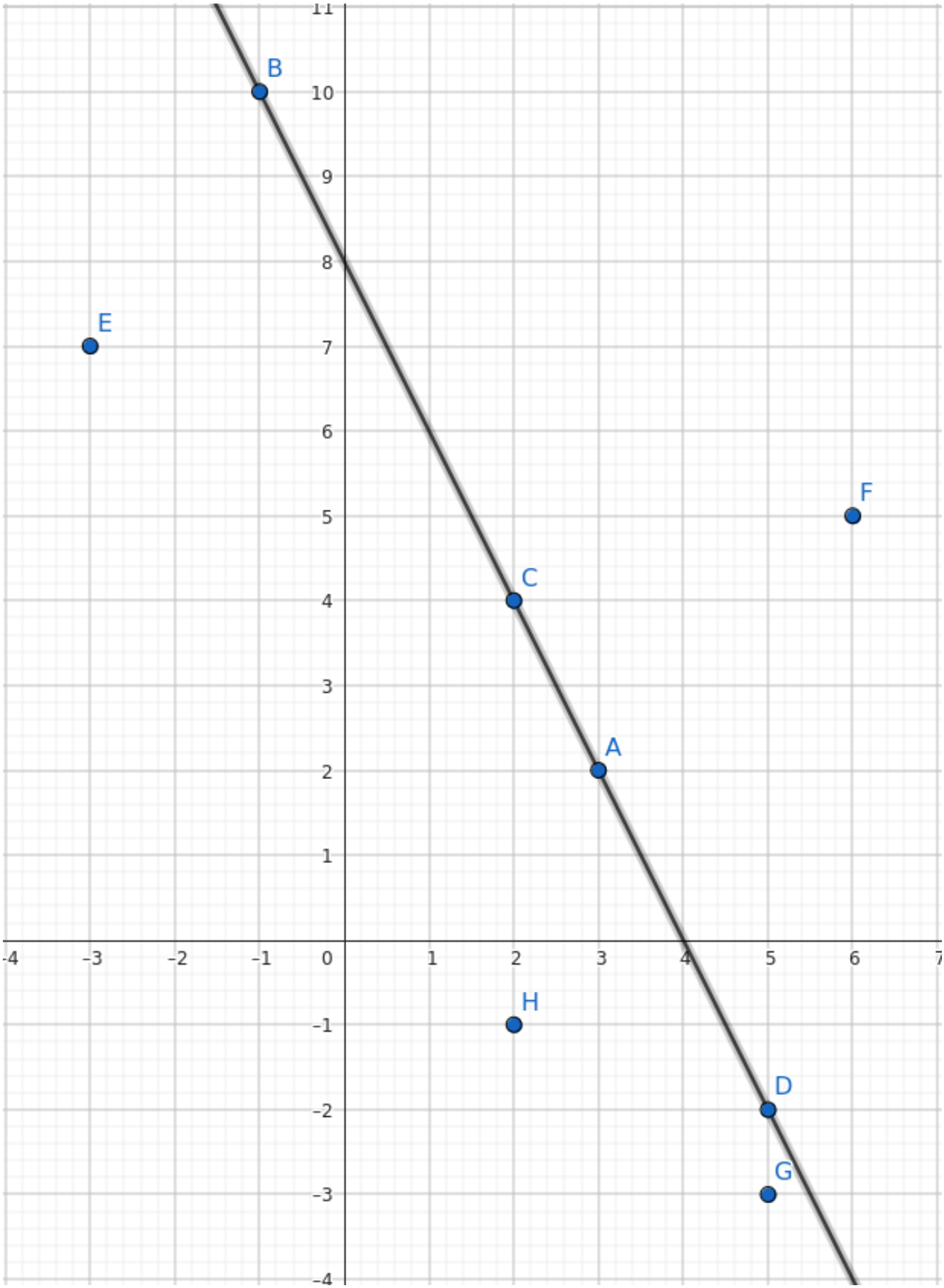
Подсказка по решению

Это задача про бинарный поиск. Именно бинарный.

Замечание

Пояснение к первому примеру:

На картинке ниже массив a содержит точки A, B, C, D ровно в таком порядке. Последняя точка в массиве имеет координаты, совпадающие с координатами точки DD, она не отмечена отдельно на картинке. Запросами являются точки E, F, G, H ровно в таком порядке. Ближайшей точкой для точки E является точка B, что видно по картинке. Ближайшей точкой для точки F является точка C, поскольку $FC = \sqrt{4^2 + 1^2} = \sqrt{17}, FA = \sqrt{3^2 + 3^2} = \sqrt{18}$. Ближайшей точкой для точки G является точка D, что видно по картинке. Точка H равноудалена от точек A и D, но точка $A = (3, 2)$ лексикографически меньше точки $D = (5, -2)$, поэтому ответом на последний запрос будет $(3, 2)$.


B
E
F
C
A
H
D
G

Задача 5G. Сила с тобой, Люк [3, 256]

Дан массив a из n чисел, нужно научиться обрабатывать запросы двух типов.

- `change (i, y)` — сделать a_i равным y.
- `int get ()` — вернуть индекс i такой, что a_i встречается в массиве наименьшее возможное число раз. Если таких i несколько, вернуть минимально возможный.

Все индексы, встречающиеся в задаче, нумеруются с нуля.

Формат входных данных

На первой строке размер массива n.
На второй сам массив — n целых чисел от 0 до 10^9-1.
На третей строке число запросов q.
Следующие q строк содержат запросы в формате "?" (`get`) и "$=$ i y" (`change`).
Ограничения: $n, q \leqslant 300\,000$.

Формат выходных данных

На каждый запрос `get` выведите на отдельной строке ответ.

Замечание

Поскольку 8.5 мегабайт нельзя прочитать из файла мгновенно (как и записать 1 мегабайт данных), используйте максимально быстрые ввод вывод.

Примеры

stdin	stdout
4 1 2 3 4 5 ? = 0 2 ? = 3 3 ?	0 2 0
6 1 1 2 1 1 0 3 ? = 5 2 ?	2 2