Задача A. Место встречи изменить нельзя

Имя входного файла:	`rendezvous.in`
Имя выходного файла:	`rendezvous.out`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Даны N точек. Найдите такие две из них, что расстояние между ними минимально.

Формат входных данных

Первая строка входного файла содержит целое число N ($2 \leqslant N \leqslant 100\,000$) — количество точек. Каждая из следующих N строк содержит пару целых чисел X и Y, разделённых пробелом, — координаты ($-1\,000\,000\,000 \leqslant X, Y \leqslant 1\,000\,000\,000$). Все точки различны.

Формат выходных данных

Единственная строка выходного файла должна содержать координаты двух выбранных точек.

Пример

rendezvous.in	rendezvous.out
4 0 0 0 1 1 1 1 0	0 0 0 1

Задача B. Соединение и разъединение

Имя входного файла:	`connect.in`
Имя выходного файла:	`connect.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	512 мегабайт

Вы когда-нибудь слышали про обход в глубину? Например, используя этот алгоритм, вы можете проверить является ли граф связным за время $O(E)$. Вы можете даже посчитать количество компонент связности за то же время.

А вы когда нибудь слышали про систему непересекающихся множеств? Используя эту структуру, вы можете быстро обрабатывать запросы "Добавить ребро в граф" и "Посчитать количество компонент связности в графе".

А вы когда-нибудь слышали о *динамической* задаче связности? В этой задаче вам необходимо обрабатывать три типа запросов:

1. Добавить ребро в граф.
2. Удалить ребро из графа.
3. Посчитать количество компонент связности в графе.

Можно считать, что граф является неориентированным. Изначально граф является пустым.

Формат входных данных

В первой строке находятся два целых числа N и K — количество вершин и количество запросов, сответственно ($1 \leqslant N \leqslant 300\,000$, $0 \leqslant K \leqslant 300\,000$). Следующие K строк содержат запросы, по одному в строке. Каждый запрос имеет один из трех типов:

1. `+` u v: Добавить ребро между вершинами u и v. Гарантируется, что такого ребра нет.
2. `-` u v: Удалить ребро между u и v. Гарантируется, что такое ребро есть.
3. `?`: Посчитать количество компонент связности в графе.

Вершины пронумерованы целыми числами от 1 до N. Во всех запросах $u \neq v$.

Формат выходных данных

Для каждого запроса типа '`?`', Выведите количество компонент связности в момент запроса.

Примеры

connect.in	connect.out
5 11 ? + 1 2 + 2 3 + 3 4 + 4 5 + 5 1 ? - 2 3 ? - 4 5 ?	5 1 1 2

Задача C. Задача для второклассника

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

Вам даны два числа. Необходимо найти их произведение.

Формат входных данных

Входные данные состоят из двух строк, на каждой из которых находится целое одно **целое** число, длина которого не превосходит двухсот пятидесяти тысяч символов.

Формат выходных данных

Выведите произведение данных чисел.

Примеры

multiply.in	multiply.out
2 2	4

Задача D. Тандемные повторы

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

Дана строка s длины n.

Тандемным повтором в ней называются два вхождения какой-либо подстроки подряд. Иными словами, тандемный повтор описывается парой индексов $i < j$ такими, что подстрока $s[i \dots j]$ — это две одинаковые строки, записанные подряд.

От вас требуется посчитать количество пар индексом $i < j$ таких, что подстрока $s[i \dots j]$ является тандемным повтором.

Формат входных данных

Во входном файле находятся не более 30 тестов. Каждый тест состоит из единственной непустой строки, состоящей из символов **A**,**C**,**G**,**T**. Длина строки не превосходит 10^5. Входной файл заканчивается строкой **0**.

Формат выходных данных

Для каждого теста выведите единственное число — количество тандемных повторов. Числа разделяйте переводами строк.

Примеры

tandems.in	tandems.out
AGGA AGAG ATTCGATTCGATTCG AAAA 0	1 1 9 4

Задача E. Дружелюбные хомячки

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

На плоскости живут n хомячков. Каждый в точке с целыми координатами. Хомячки дружат, если существует прямоугольник со сторонами, параллельными осям координат, содержащий этих двух хомячков и не содержащий никаких других.

Прямоугольник содержит хомячка, если точка, в которой он живет, лежит внутри прямоугольника или на его границе.

Сколько пар хомячков дружат?

Формат входных данных

На первой строке число n, $1 \leqslant n \leqslant 100\,000$.

Следующие n строк содержат по два целых числа — координаты точек, в которых живут хомячки.

Все точки различны, а координаты целые, по модулю не превосходят 10^9.

Формат выходных данных

Выведите одно целое число — количество пар дружащих хомячков.

Примеры

стандартный ввод	стандартный вывод
5 0 0 0 2 2 0 2 2 1 1	8

Задача F. Перестановки strike back

Имя входного файла:	`permutation2.in`
Имя выходного файла:	`permutation2.out`
Ограничение по времени:	5 секунд
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 большим, 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 количество 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 G. Прибавления на отрезках

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Когда-то Костя решал учебную задачу о поиске максимума с прибавлением на отрезке. Формально задача звучала так:

«Дан массив длины n, изначально состоящий из нулей. Элементы массива пронумерованы от 1 до n. К массиву было применено q операций. i-я операция задается тремя целыми числами l_i, r_i и x_i ($1 \leqslant l_i \leqslant r_i \leqslant n$), ($1 \leqslant x_i \leqslant n$) и означает, что к элементам с номерами $l_i, l_i + 1, \ldots, r_i$ прибавили число x_i. Требуется найти максимум в массиве после применения всех этих операций.»

Много лет спустя Костя нашел в своем Github-профиле решение этой задачи. Костя решил, что хочет улучшить свой старый код. В том числе, он решил, что в целях экономии памяти уберет запретную строчку кода, которая превращает все 32-битные числа в 64-битные. Однако, он хочет удостовериться, что решение все еще будет корректным.

Вычисляя ограничения на максимум в рамках данной задачи, он задумался: «интересно, а какие значения может принять максимум в массиве после применения некоторого подмножества данных операций?».

Помогите Косте, найдите все такие целые числа y от 1 до n, что после применения некоторого (возможно, пустого) подмножества данных операций максимум в массиве равен y.

Формат входных данных

В первой строке находятся два целых числа n и q ($1 \leqslant n, q \leqslant 10^4$) — длина массива и количество запросов в исходной задаче.

В следующих q строках описаны запросы, по одному в строке. i-я из этих строк содержит три целых числа l_i, r_i и x_i ($1 \leqslant l_i \leqslant r_i \leqslant n$, $1 \leqslant x_i \leqslant n$), что обозначает запрос на добавление числа x_i на отрезке с l_i-го по r_i-й элемент включительно.

Формат выходных данных

В первую строку выведите единственное число k, обозначающее количество возможных целых чисел от 1 до n, которым может быть равен максимум в массиве после применения некоторого (возможно, пустого) подмножества данных операций.

В следующей строке выведите через пробел все k чисел от 1 до n — возможные значения максимума. Выводите эти числа **в возрастающем порядке**.

Примеры

стандартный ввод	стандартный вывод
4 3 1 3 1 2 4 2 3 4 4	4 1 2 3 4
7 2 1 5 1 3 7 2	3 1 2 3
10 3 1 1 2 1 1 3 1 1 6	6 2 3 5 6 8 9

Замечание

Если в первом тестовом примере оставить только первый запрос, то максимум будет равен 1. Если оставить только второй запрос, то максимум будет равен 2. Если оставить первые два запроса, то максимум будет равен 3. Если оставить только третий запрос, то максимум будет равен 4. Но

если оставить третий запрос и еще какой-то, максимум будет больше n, поэтому его выводить не требуется.

Во втором тестовом примере, оставив только первый запрос, можно получить 1. Оставив только второй, можно получить 2. А если оставить все запросы, максимум будет равен 3.

В третьем тестовом примере можно получить максимумы так:

- Можно получить максимум 2 оставив запросы: (1).
- Можно получить максимум 3 оставив запросы: (2).
- Можно получить максимум 5 оставив запросы: $(1, 2)$.
- Можно получить максимум 6 оставив запросы: (3).
- Можно получить максимум 8 оставив запросы: $(1, 3)$.
- Можно получить максимум 9 оставив запросы: $(2, 3)$.