,

Задача A. И снова сумма...

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

Реализуйте структуру данных, которая поддерживает множество S целых чисел, с котором разрешается производить следующие операции:

- $add(i)$ — добавить в множество S число i (если он там уже есть, то множество не меняется);
- $sum(l, r)$ — вывести сумму всех элементов x из S, которые удовлетворяют неравенству $l \leqslant x \leqslant r$.

Формат входных данных

Исходно множество S пусто. Первая строка входного файла содержит n — количество операций ($1 \leqslant n \leqslant 300\,000$). Следующие n строк содержат операции. Каждая операция имеет вид либо «`+` i», либо «`?` l r». Операция «`?` l r» задает запрос $sum(l, r)$.

Если операция «`+` i» идет во входном файле в начале или после другой операции «`+`», то она задает операцию $add(i)$. Если же она идет после запроса «`?`», и результат этого запроса был y, то выполняется операция $add((i + y) \bmod 10^9)$.

Во всех запросах и операциях добавления параметры лежат в интервале от 0 до 10^9.

Формат выходных данных

Для каждого запроса выведите одно число — ответ на запрос.

Пример

стандартный ввод	стандартный вывод
6 + 1 + 3 + 3 ? 2 4 + 1 ? 2 4	3 7

,

Задача B. Следующий

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Реализуйте структуру данных, которая поддерживает множество S целых чисел, с котором разрешается производить следующие операции:

1. `add(i)` – добавить в множество S число i (если он там уже есть, то множество не меняется)
2. `next(i)` – вывести минимальный элемент множества, не меньший i. Если искомый элемент в структуре отсутствует, необходимо вывести -1.

Формат входных данных

Исходно множество S пусто. Первая строка входного файла содержит n – количество операций ($1 \leqslant n \leqslant 300\,000$). Следующие n строк содержат операции. Каждая операция имеет вид либо `«+ i»`, либо `«? i»`. Операция `«? i»` задает запрос `next(i)`.

Если операция `«+ i»` идет во входном файле в начале или после другой операции `«+»`, то она задает операцию `add(i)`. Если же она идет после запроса `«?»`, и результат этого запроса был y, то выполняется операция $add((i + y) \mod 10^9)$.

Во всех запросах и операциях добавления параметры лежат в интервале от 0 до 10^9.

Формат выходных данных

Для каждого запроса выведите одно число – ответ на запрос.

Пример

стандартный ввод	стандартный вывод
6 + 1 + 3 + 3 ? 2 + 1 ? 4	3 4

Замечание

В этой задаче запрещено использовать STL.

,

Задача C. НОД 2010

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Вам предложили работу в секретном проекте Агентства Федеральной Безопасности под кодовым названием «НОД 2010». Основным объектом исследования является набор целых положительных чисел. Вы должны понять, как будет изменяться наибольший общий делитель всех чисел этого набора при добавлении в него новых чисел или удалении лежащих там чисел. В начале эксперимента набор чисел пуст.

Формат входных данных

В первой строке записано целое число q ($1 \leqslant q \leqslant 10^5$) — количество операций с набором.

Каждая из следующих q строк имеет вид «`+ x`» или «`- x`». В первом случае число x добавляется в набор, а во втором случае — удаляется из него. Число x целое, положительное и не превосходит 10^9. Гарантируется, что из набора будут удаляться только числа, которые в нём лежат.

Формат выходных данных

Выведите наибольший общий делитель всех чисел набора после каждой описанной операции. Согласно распоряжению 190р, наибольшим общим делителем пустого набора является единица.

Пример

стандартный ввод	стандартный вывод
5 + 8 + 6 + 8 - 8 - 8	8 2 2 2 6

,

Задача D. Вперёд!

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

256 мегабайт

Капрал Дукар любит раздавать приказы своей роте. Самый любимый его приказ — «Вперёд!». Капрал строит солдат в ряд и отдаёт некоторое количество приказов, каждый из которых звучит так: «Рядовые с l_i по l_j — вперёд!»

Перед тем, как Дукар отдал первый приказ, солдаты были пронумерованы от 1 до n слева направо. Услышав приказ «Рядовые с l_i по l_j — вперёд!», солдаты, стоящие на местах с l_i по l_j включительно, продвигаются в начало ряда в том же порядке, в котором были.

Например, если в какой-то момент солдаты стоят в порядке $2, 3, 6, 1, 5, 4$, то после приказа «Рядовые с 2 по 4 — вперёд!», порядок будет таким: $3, 6, 1, 2, 5, 4$. А если потом Капрал вышлет вперёд солдат с 3 по 4, то порядок будет уже таким: $1, 2, 3, 6, 5, 4$.

Вам дана последовательность приказов Капрала. Найдите порядок, в котором будут стоять солдаты после исполнения всех приказов.

Формат входных данных

В первой строке входного файла указаны числа n и m ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 100\,000$) — число солдат и число приказов. Следующие m строк содержат приказы в виде двух целых чисел: l_i и r_i ($1 \leqslant l_i \leqslant r_i \leqslant n$).

Формат выходных данных

Выведите в выходной файл n целых чисел — порядок, в котором будут стоять солдаты после исполнения всех приказов.

Пример

стандартный ввод	стандартный вывод
6 3 2 4 3 5 2 2	1 4 5 2 3 6

,

Задача E. K-ый максимум

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	0.5 секунд
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.

Пример

стандартный ввод	стандартный вывод
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

,

Задача F. Переворот

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	2 секунды
Ограничение по памяти:	64 мегабайта

Дан массив. Надо научиться обрабатывать два типа запросов.

- `1 L R` - перевернуть отрезок [L, R]
- `2 L R` - найти минимум на отрезке [L, R]

Формат входных данных

Первая строка файла содержит два числа n, m. ($1 \leqslant n, m \leqslant 10^5$) Во второй строке находится n чисел a_i ($1 \leqslant a_i \leqslant 10^9$)- исходный массив. Остальные m строк содержат запросы, в формате описанном в условии. Для чисел L,R выполняется ограничение ($1 \leqslant L \leqslant R \leqslant n$).

Формат выходных данных

На каждый запрос типа 2, во входной файл выведите ответ на него, в отдельной строке.

Пример

стандартный ввод	стандартный вывод
10 7 5 3 2 3 12 6 7 5 10 12 2 4 9 1 4 6 2 1 8 1 1 8 1 8 9 2 1 7 2 3 6	3 2 2 2

,

Задача G. Очередная

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

Изначально вам дана перестановка чисел от 1 до N. Вам поступают запросы двух видов:

- 1 l_1 r_1 l_2 r_2 для выполнения требуется взять два подмассива нашей перестановки с границами $[l_1, r_1]$ и $[l_2, r_2]$ и поменять местами содержимое подмассивов друг с другом.
- 2 x найти место в перестановке, где находится число x и вывести 3 следующих за ним числа

Формат входных данных

В первой строке находится два числа N и Q — размер перестановки и общее количество запросов ($2 \leqslant N \leqslant 10000$, $1 \leqslant Q \leqslant 200000$). Во второй строке — перестановка чисел от одного до N. В следующих Q строках описаны запросы в виде либо 1 l_1 r_1 l_2 r_2($1 \leqslant l_1 \leqslant r_1 < l_2 \leqslant r_2 \leqslant N$, $r_1 - l_1 = r_2 - l_2$) либо 2 x ($1 \leqslant x \leqslant N$).

Формат выходных данных

Для каждого запроса второго типа выведите три числа — следующие числа за заданным, либо -1, если какого-то числа нет.

Пример

стандартный ввод	стандартный вывод
6 6 1 2 3 4 5 6 2 4 1 1 2 4 5 2 4 2 1 1 1 3 4 6 2 1	5 6 -1 5 3 1 2 6 -1 2 6 4

,

Задача H. Своппер

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Современные компьютеры зацикливаются
в десятки раз эффективнее человека

Рекламный проспект OS Vista-N

Перед возвращением в штаб-квартиру корпорации Аазу и Скиву пришлось заполнить на местной таможне декларацию о доходах за время визита. Получилась довольно внушительная последовательность чисел. Обработка этой последовательности заняла весьма долгое время.

— Своппер кривой, — со знанием дела сказал таможенник.

— А что такое своппер? — спросил любопытный Скив.

Ааз объяснил, что своппер — это структура данных, которая умеет делать следующее.

- Взять отрезок чётной длины от x до y и поменять местами число x с $x+1$, $x+2$ с $x+3$, и т.д.
- Посчитать сумму чисел на произвольном отрезке от a до b.

Учитывая, что обсчёт может затянуться надолго, корпорация «МИФ» попросила Вас решить проблему со своппером и промоделировать ЭТО эффективно.

Формат входных данных

Во входном файле заданы один или несколько тестов. В первой строке каждого теста записаны число N — длина последовательности и число M — число операций ($1 \leqslant N, M \leqslant 100\,000$). Во второй строке теста содержится N целых чисел, не превосходящих 10^6 по модулю — сама последовательность. Далее следуют M строк — запросы в формате `1` x_i y_i — запрос первого типа, и `2` a_i b_i — запрос второго типа. Сумма всех N и M по всему файлу не превосходит 200 000. Файл завершается строкой из двух нулей. Гарантируется, что $x_i < y_i$, а $a_i \leqslant b_i$.

Формат выходных данных

Для каждого теста выведите ответы на запросы второго типа, как показано в примере. Разделяйте ответы на тесты пустой строкой.

Пример

стандартный ввод	стандартный вывод
5 5 1 2 3 4 5 1 2 5 2 2 4 1 1 4 2 1 3 2 4 4 0 0	Swapper 1: 10 9 2

,

Задача I. Логирование

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Лиза пишет программу для анализа логов дистрибутивных компьютерных систем. Когда узел становится онлайн, это может протолкнуть партию событий логов в прошлом. И наоборот, когда он переходит в автономный режим некоторые записи логов могут исчезнуть.

Чтобы обеспечить стабильность и доступность приложения Лизе необходимо контролировать число различных событий в сегментах лога. Она будет разбираться с распределительной частью, в то время как вы должны реализовать локальный.

Изначально список логов пуст, и вы должны поддерживать следующие операции:

- **insert** `<index>` `<number>` `<type>` — добавить `<number>` событий типа `<type>` перед событием с индексом `<index>`. Все события, у которых индекс больше или равен `<index>` нумеруются заново.
- **remove** `<index>` `<number>` — удалить `<number>` элементов, начиная с элемента под индексом `<index>`.
- **query** `<index_1>` `<index_2>` — вывести количество различных типов событий на отрезке с `<index_1>` до `<index_2>` включительно.

События нумеруются с 1. Тип событий представляется одним латинским символом.

Формат входных данных

В первой строке входного файла содержится единственное целое число n — количество операций ($1 \leqslant n \leqslant 30\,000$). Следующие по n строк содержат описание запросов.

Описание операции начинается с типа операции: '`+`' для добавления, '`-`' для удаления и '`?`' для запроса. Дальше следует аргументы запроса, описанные в условиях выше.

Все запросы валидны, элементы с такими индексами существуют, нет запросов на удаление несуществующих элементов.

Количество запросов добавления и удаления не превышает 10 000.

Типы событий представлены в виде строчной буквы латинского алфавита.

Формат выходных данных

Для каждого запроса **query** выведите одно целое число — количество различных типов событий на отрезке `<index_1>`, `<index_2>` включительно.

Пример

стандартный ввод	стандартный вывод
8 + 1 4 w + 3 3 o ? 2 3 - 2 2 ? 2 3 + 2 2 t ? 1 6 - 1 6	2 1 3

Замечание

Пояснение к примеру:

1. `wwww`

2. `wwoooww`
3. `w[wo]ooww` : 2 различных события
4. `wooww`
5. `w[oo]ww` : 1 событие
6. `wttooww`
7. `[wttoow]w` : 3 различных события
8. `w`

Задача J. Гроб гроб кладбище treap

Имя входного файла:	`стандартный ввод`
Имя выходного файла:	`стандартный вывод`
Ограничение по времени:	1 секунда
Ограничение по памяти:	256 мегабайт

Дан массив целых чисел. Необходимо реализовать структуру данных, в которой требуется за $O(\log n)$ выполнять запросы:

1. сумма на подотрезке $[L, R]$ (в задаче принята 0-индексация);
2. вставить элемент x в позицию pos (т.е. в результате вставки, элемент x должен оказаться pos-м);
3. удалить элемент x, находящийся на позиции i;
4. присвоить элемент x на подотрезке $[L, R]$;
5. прибавить число x на подотрезке $[L, R]$;
6. $next\_permutation$ на подотрезке $[L, R]$;
7. $prev\_permutation$ на подотрезке $[L, R]$.

$next\_permutation$ и $prev\_permutation$ должны работать так же, как одноименные STL-алгоритмы; В частности, $next\_permutation([4, 3, 2, 1])$ есть $[1, 2, 3, 4]$, а не $[4, 3, 2, 1]$; Аналогично, $prev\_permutation([1, 2, 2, 2, 3, 3, 4]) = [4, 3, 3, 2, 2, 2, 1]$.

Формат входных данных

В первой строке записано число $n (1 \leqslant n \leqslant 3 \cdot 10^4)$ — количество элементов в массиве. Во второй строке записано n чисел, не превосходящих по модулю $3 \cdot 10^4$ — исходные значения элементов массива.

В третьей строке записано число $q (1 \leqslant q \leqslant 10^5)$ - количество запросов. В последующих строках записаны сами запросы, по одному на каждой строке.

Запросы задаются в следующем формате:

- 1 L R ($0 \leqslant l \leqslant r < arraySize$ - найти сумму всех чисел массива на отрезке [l, r];
- 2 x pos ($|x| \leqslant 3 \cdot 10^4, 0 \leqslant pos \leqslant arraySize$): вставить элемент x на позицию pos;
- 3 pos ($0 \leqslant pos < arraySize$): удалить элемент, находящийся на позиции pos;
- 4 x L R ($|x| \leqslant 3 \cdot 10^4, 0 \leqslant l \leqslant r < arraySize$): присвоить элементам на подотрезке $[L, R]$ значение x;
- 5 x L R ($|x| \leqslant 3 \cdot 10^4, 0 \leqslant l \leqslant r < arraySize$): прибавить к элементам на подотрезке $[L, R]$ число x;
- 6 L R: выполнить $next\_permutation$ на подотрезке $[L, R]$;
- 7 L R: выполнить $prev\_permutation$ на подотрезке $[L, R]$.

В приведенном описании $arraySize$ — размер массива на момент запроса. Иными словами вам гарантируется, что все запросы корректные.

Формат выходных данных

Для каждого запроса типа 1 выведите соответствующую сумму в отдельной строке. По выполнении всех запросов, выведите итоговые значения элементов массива также в отдельной строке.

Пример

стандартный ввод	стандартный вывод
7 1 2 3 4 5 6 7 8 4 5 1 3 2 3 3 5 2 0 4 7 0 6 6 0 3 3 2 1 1 5 1 0 6	28 40 5 3 7 6 7 5 7