Задача MoveToFront. В начало строя!

Имя входного файла:	`movetofront.in`
Имя выходного файла:	`movetofront.out`
Ограничение по времени:	4 секунды
Ограничение по памяти:	64 мебибайта

Капрал Питуца любит командовать своим отрядом. Его любимый приказ «в начало строя». Он выстраивает свой отряд в шеренгу и оглашает последовательность приказов. Каждая приказ имеет вид «Солдаты с l_i по r_i — в начало строя!»

Пронумеруем солдат в начальном положении с 1 до n, слева направо. Приказ «Солдаты с l_i по r_i — в начало строя!» означает, что солдаты, стоящие с l_i по r_i включительно перемещаются в начало строя, сохраняя относительный порядок.

Например, если в некотрый момент солдаты стоят в порядке $2, 3, 6, 1, 5, 4$, после приказа: «Солдаты с 2 по 4 — в начало строя!» порядок будет $3, 6, 1, 2, 5, 4$.

По данной последовательности приказов найти конечный порядок солдат в строю.

Формат входного файла

В первой строке два целых числа n and m ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 100\,000$) — количество солдат и количество приказов. Следующие m строк содержат по два целых числа l_i и r_i ($1 \leqslant l_i \leqslant r_i \leqslant n$).

Формат выходного файла

Выведите n целых чисел — порядок солдат в конечном положении после выполнения всех приказов.

Пример

movetofront.in	movetofront.out
6 3 2 4 3 5 2 2	1 4 5 2 3 6

Задача RMQ. Range Minimum Query

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

Компания *Giggle* открывает свой новый офис в Судиславле, и вы приглашены на собеседование. Ваша задача — решить поставленную задачу[1].

Вам нужно создать структуру данных, которая представляет из себя массив целых чисел. Изначально массив пуст. Вам нужно поддерживать две операции:

- запрос: «`? i j`» — возвращает минимальный элемент между i-ым и j-м, включительно;
- изменение: «`+ i x`» — добавить элемент x после i-го элемента списка. Если $i = 0$, то элемент добавляется в начало массива.

Конечно, эта структура должна быть достаточно хорошей.

Формат входного файла

Первая строка входного файла содержит единственное целое число n — число операций над массивом ($1 \leqslant n \leqslant 200\,000$). Следующие n строк описывают сами операции. Все операции добавления являются корректными. Все числа, хранящиеся в массиве, по модулю не превосходят 10^9.

Формат выходного файла

Для каждой операции в отдельной строке выведите её результат.

[1] Капитан Очевидность намекает

Пример

rmq.in	rmq.out
8 + 0 5 + 1 3 + 1 4 ? 1 2 + 0 2 ? 2 4 + 4 1 ? 3 5	4 3 1

Нижеследующая таблица показывает процесс изменения массива из примера.

Операция	**Массив после её выполнения**
изначально	*пуст*
+ 0 5	5
+ 1 3	5, 3
+ 1 4	5, 4, 3
+ 0 2	2, 5, 4, 3
+ 4 1	2, 5, 4, 3, 1

Задача Swapper. Своппер

Имя входного файла:	`swapper.in`
Имя выходного файла:	`swapper.out`
Ограничение по времени:	3 секунды
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 входного файла

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 выходного файла

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

Пример

swapper.in	swapper.out
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