Задача A. Вперёд!

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

Примеры

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

Задача B. 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.

Примеры

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 C. Переворот

Имя входного файла:	`reverse.in`
Имя выходного файла:	`reverse.out`
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.

Примеры

reverse.in	reverse.out
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 D. Своппер

Имя входного файла:	`swapper.in`
Имя выходного файла:	`swapper.out`
Ограничение по времени:	4 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 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.

Примеры

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

Задача E. Очередь

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

В зимний холодный вечер наш герой Вася стоял в очереди на вокзале за билетом на финал чемпионата Codeforces. Как обычно это бывает, кассир, сказав, что он отлучится на 5 минут, ушел на целый час... Тогда Вася, чтобы не скучать в это время, стал изучать такой механизм, как очередь. Наблюдения потрясли Васю.

Каждый человек характеризуется числами a_i — важностью его дел (чем это число больше, тем важнее его дело), и числом c_i — характеристика его совести. Числа a_i образуют перестановку чисел от 1 до n.

Пусть на данный момент очередь состоит из $n-1$ людей. Рассмотрим, как ведет себя пришедший номер n. Первым делом он встает в конец текущей очереди, а дальше поступает следующим образом: если у человека, который стоит перед ним важность дел a_i меньше чем a_n, то они меняются местами (выглядит это так: человек номер n спрашивает у предыдущего: «Эээ... простите, пожалуйста, у меня очень важное дело... можете меня пропустить вперед?»), затем он опять спрашивает у впереди стоящего человека, и так далее. Если же a_i больше чем a_n, то продвижение вперед заканчивается. Но такую операцию обмена человек номер n может совершить не более, чем c_n раз.

В нашей задаче будем считать, что к моменту, когда человек номер n придет в очередь, процесс обменов в очереди из $n-1$ людей уже закончится. Если обмен возможен, то он обязательно происходит.

Ваша задача — помочь Васе промоделировать описанный процесс и найти, в каком порядке пришедшие люди будут располагаться в очереди после окончания всех обменов.

Формат входных данных

В первой строке входных данных находится целое число n — количество людей, пришедших в данную очередь ($1 \leqslant n \leqslant 10^5$). Далее в n строках заданы описания людей в порядке их прихода — целые числа a_i и c_i ($1 \leqslant a_i \leqslant n$, $0 \leqslant c_i \leqslant n$), записанные через пробел. Каждое описание находится на отдельной строке. Все a_i различны.

Формат выходных данных

Выведите перестановку чисел от 1 до n, означающую образованную по описанным правилам очередь в порядке от начала к концу. В этой последовательности i-ое число означает номер человека, который будет стоять в очереди на месте номер i после завершения процесса обменов. Люди нумеруются с 1 в порядке, в котором они заданы во входных данных. Числа разделяйте пробелом.

Примеры

queue.in	queue.out
2 1 0 2 1	2 1
3 1 3 2 3 3 3	3 2 1
5 2 3 1 4 4 3 3 1 5 2	3 1 5 4 2

Задача F. Звезды

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

Вася любит наблюдать за звездами. Но следить за всем небом сразу ему тяжело. Поэтому он наблюдает только за частью пространства, ограниченной кубом размером $n \times n \times n$. Этот куб поделен на маленькие кубики размером $1 \times 1 \times 1$. Во время его наблюдений могут происходить следующие события:

1. В каком-то кубике появляются или исчезают несколько звезд.
2. К нему может заглянуть его друг Петя и поинтересоваться, сколько видно звезд в части пространства, состоящей из нескольких кубиков.

Формат входных данных

Первая строка входного файла содержит натуральное число $1 \leqslant n \leqslant 128$. Координаты кубиков — целые числа от 0 до $n - 1$. Далее следуют записи о происходивших событиях по одной в строке. В начале строки записано число m. Если m равно:

1, то за ним следуют 4 числа — x, y, z ($0 \leqslant x, y, z < N$) и k ($-20000 \leqslant k \leqslant 20000$) — координаты кубика и величина, на которую в нем изменилось количество видимых звезд;

2, то за ним следуют 6 чисел — x_1, y_1, z_1, x_2, y_2, z_2 ($0 \leqslant x_1 \leqslant x_2 < N$, $0 \leqslant y_1 \leqslant y_2 < N$, $0 \leqslant z_1 \leqslant z_2 < N$), которые означают, что Петя попросил подсчитать количество звезд в кубиках (x, y, z) из области: $x_1 \leqslant x \leqslant x_2$, $y_1 \leqslant y \leqslant y_2$, $z_1 \leqslant z \leqslant z_2$;

3, то это означает, что Васе надоело наблюдать за звездами и отвечать на вопросы Пети. Эта запись встречается во входном файле только один раз и будет последней.

Количество записей во входном файле не больше 100 002.

Формат выходных данных

Для каждого Петиного вопроса выведите искомое количество звезд.

Примеры

stars.in	stars.out
2 2 1 1 1 1 1 1 1 0 0 0 1 1 0 1 0 3 2 0 0 0 0 0 0 2 0 0 0 0 1 0 1 0 1 0 -2 2 0 0 0 1 1 1 3	0 1 4 2