

## Задача А. Range Variation Query

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

В начальный момент времени последовательность  $a_n$  задана следующей формулой:  $a_n = n^2 \bmod 12345 + n^3 \bmod 23456$ .

Требуется много раз отвечать на запросы следующего вида:

- найти разность между максимальным и минимальным значением среди элементов  $a_i, a_{i+1}, \dots, a_j$ ;
- присвоить элементу  $a_i$  значение  $j$ .

### Формат входного файла

Первая строка входного файла содержит натуральное число  $k$  — количество запросов ( $k \leq 100\,000$ ). Следующие  $k$  строк содержат запросы, по одному на строке. Запрос номер  $i$  описывается двумя целыми числами  $x_i, y_i$ .

Если  $x_i > 0$ , то требуется найти разность между максимальным и минимальным значением среди элементов  $a_{x_i}, \dots, a_{y_i}$ . При этом  $1 \leq x_i \leq y_i \leq 100\,000$ .

Если  $x_i < 0$ , то требуется присвоить элементу  $a_{|x_i|}$  значение  $y_i$ . При этом  $-100\,000 \leq x_i \leq -1$  и  $|y_i| \leq 100\,000$ .

### Формат выходного файла

Для каждого запроса первого типа в выходной файл требуется вывести одну строку, содержащую разность между максимальным и минимальным значением на соответствующем отрезке.

### Пример

rvq.in	rvq.out
7	34
1 3	68
2 4	250
-2 -100	234
1 5	1
8 9	
-3 -101	
2 3	

## Задача В. Фонари

Имя входного файла: `light.in`  
Имя выходного файла: `light.out`  
Ограничение по времени: 1 секунда  
Ограничение по памяти: 64 мегабайта

Администрация Ханты-Мансийска установила вдоль улицы Мира  $N$  фонарей. Лампочки в фонарях могут перегорать, и, хотя в Ханты-Мансийске есть электрик, он может заболеть и не успеть заменить перегоревшую лампочку на новую до приезда очередной краевой комиссии.

Каждая посещающая город комиссия считает своим долгом проинспектировать состояние фонарей, освещающих улицу Мира, но улица эта очень длинная, поэтому каждый раз членов комиссии подвозят на машине к какому-нибудь фонарю, они выходят, идут пешком до другого фонаря, осматривая по пути все фонари, там садятся в машину и уезжают.

Если они замечают хотя бы один перегоревший фонарь, свет на всей дороге выключают и школьники ближайшим вечером не могут вернуться из Лицея домой.

По информации о происходящих событиях для каждого приезда комиссии выведите результат осмотра фонарей.

### Формат входного файла

В первой строке входного файла записаны два числа  $N$  и  $k$  ( $1 \leq N, k \leq 100\,000$ ) — количество фонарей и количество событий. Далее следуют  $k$  строк, описывающих события. События описываются следующим образом:

- перегорание фонаря задаётся двумя числами, первое из которых равно  $-1$ , а второе задаёт номер перегоревшего фонаря (от 1 до  $N$ );
- замена лампочки электриком задаётся двумя числами, первое из которых равно 1, а второе задаёт номер заменённой лампочки (от 1 до  $N$ );
- приезд комиссии задается тремя числами, первое из которых равно 0, а два других ( $a$  и  $b$ ) задают отрезок дороги, по которому проходит комиссия. А именно, члены комиссии видят все фонари с номерами от  $a$  до  $b$  включительно и только их. Гарантируется, что  $1 \leq a \leq b \leq N$ .

Изначально все фонари исправны.

### Формат выходного файла

Для каждого приезда комиссии выведите в выходной файл одну строку. А именно, если она не заметит ни одного перегоревшего фонаря, то выведите «PASSED», иначе — «PENALTY».

### Пример

light.in	light.out
5 6	PASSED
-1 2	PENALTY
0 1 1	PASSED
0 2 5	
-1 4	
1 2	
0 2 3	

## Задача С. Ближайшее большее число справа

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

Дан массив  $a$  из  $n$  чисел. Нужно обрабатывать запросы:

0. `set(i, x)` – присвоить новое значение элементу массива  $a[i] = x$ ;
1. `get(i, x)` – найти  $\min k: k \geq i$  и  $a_k \geq x$ .

### Формат входного файла

Первая строка входных данных содержит два числа: длину массива  $n$  и количество запросов  $m$  ( $1 \leq n, m \leq 200\,000$ ).

Во второй строке записаны  $n$  целых чисел – элементы массива  $a$  ( $0 \leq a_i \leq 200\,000$ ).

Следующие  $m$  строк содержат запросы, каждый запрос содержит три числа  $t, i, x$ . Первое число  $t$  равно 0 или 1 – тип запроса.  $t = 0$  означает запрос типа `set`,  $t = 1$  соответствует запросу типа `get`,  $1 \leq i \leq n$ ,  $0 \leq x \leq 200\,000$ . Элементы массива нумеруются с единицы.

### Формат выходного файла

На каждой запрос типа `get` на отдельной строке выведите соответствующее значение  $k$ . Если такого  $k$  не существует, выведите  $-1$ .

### Примеры

<code>nearandmore.in</code>	<code>nearandmore.out</code>
4 5	1
1 2 3 4	3
1 1 1	-1
1 1 3	2
1 1 5	
0 2 3	
1 1 3	

## Задача D. Окна

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

На экране расположены прямоугольные окна, возможно, каким-то образом перекрывающиеся (со сторонами, параллельными осям координат). Вам необходимо найти точку, которая покрыта наибольшим числом из них.

### Формат входного файла

В первой строке входного файла записано число окон  $n$  ( $1 \leq n \leq 50\,000$ ). Следующие  $n$  строк содержат координаты окон  $x_{(1,i)}$ ,  $y_{(1,i)}$ ,  $x_{(2,i)}$ ,  $y_{(2,i)}$ , где  $(x_{(1,i)}, y_{(1,i)})$  — координаты левого верхнего угла  $i$ -го окна, а  $(x_{(2,i)}, y_{(2,i)})$  — правого нижнего (на экране компьютера  $y$  растет сверху вниз, а  $x$  — слева направо). Все координаты — целые числа, по модулю не превосходящие  $2 \cdot 10^5$ .

### Формат выходного файла

В первой строке выходного файла выведите максимальное число окон, покрывающих какую-либо из точек в данной конфигурации. Во второй строке выведите два целых числа, разделенные пробелом — координаты точки, покрытой максимальным числом окон. Окна считаются замкнутыми, т. е. покрывающими свои граничные точки.

### Пример

windows.in	windows.out
2	2
0 0 3 3	3 2
1 1 4 4	