

## Задача А. Нолики

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

Дедус любит давать своим ученикам сложные задачки. На этот раз он придумал такую задачу: Рейтинг всех его учеников записан в массив  $A$ . Запросы Дедуса таковы:

1. Изменить рейтинг  $i$ -го ученика на число  $x$
2. Найти максимальную последовательность подряд идущих ноликов в массиве  $A$  на отрезке  $[l, r]$ .

Помогите бедным фиксикам избежать зверского наказания за нерешение задачи на этот раз.

### Формат входных данных

В первой строке входного файла записано число  $N$  ( $1 \leq N \leq 500\,000$ ) – количество учеников. Во второй строке записано  $N$  чисел – их рейтинги, числа по модулю не превосходящие 1000 (по количеству задач, которые ученик решил или не решил за время обучения). В третьей строке записано число  $M$  ( $1 \leq M \leq 50\,000$ ) – количество запросов. Каждая из следующих  $M$  строк содержит описание запросов:

«UPDATE  $i$   $x$ » – обновить  $i$ -ый элемент массива значением  $x$  ( $1 \leq i \leq N$ ,  $|x| \leq 1000$ )

«QUERY  $l$   $r$ » – найти длину максимальной последовательности из нулей на отрезке с  $l$  по  $r$ . ( $1 \leq l \leq r \leq N$ )

### Формат выходных данных

В выходной файл выведите ответы на запросы «QUERY» в том же порядке, что и во входном файле

### Пример

стандартный ввод	стандартный вывод
5	2
328 0 0 0 0	1
5	1
QUERY 1 3	
UPDATE 2 832	
QUERY 3 3	
QUERY 2 3	
UPDATE 2 0	

## Задача В. Вика и отрезки

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

У Вики есть бесконечный лист клетчатой бумаги. Изначально каждая клетка белого цвета. Она ввела на этом листе двумерную систему координат и нарисовала  $n$  чёрных горизонтальных и вертикальных отрезков, параллельных осям координат и шириной в одну клетку. Таким образом, каждый отрезок является множеством соседних клеток, расположенных в одной строке или в одном столбце.

Перед вами стоит задача посчитать число закрашенных клеток. Если клетка была покрашена более одного раза, в ответе она должна быть посчитана ровно один раз.

### Формат входных данных

В первой строке входных данных следует целое положительное число  $n$  ( $1 \leq n \leq 100\,000$ ) – количество отрезков, нарисованных Викой.

В следующих  $n$  строках заданы по четыре целых числа  $x_1, y_1, x_2, y_2$  ( $-10^9 \leq x_1, y_1, x_2, y_2 \leq 10^9$ ) – координаты концов отрезков, нарисованных Викой. Гарантируется, что все отрезки параллельны осям координат. Отрезки могут пересекаться, накладываться друг на друга и даже полностью совпадать.

### Формат выходных данных

Выведите количество закрашенных Викой клеток. Если клетка была покрашена более одного раза, то в ответе она должна быть посчитана ровно один раз.

### Примеры

стандартный ввод	стандартный вывод
3 0 1 2 1 1 4 1 2 0 3 2 3	8
4 -2 -1 2 -1 2 1 -2 1 -1 -2 -1 2 1 2 1 -2	16

## Задача С. Сережа и скобочки

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

У Сережи есть строка  $s$  длины  $n$ , состоящая из символов «(» и «)».

Сереже нужно ответить на  $m$  запросов, каждый из которых характеризуется двумя целыми числами  $l_i, r_i$ . Ответом на  $i$ -ый запрос является длина наибольшей правильной скобочной подпоследовательности последовательности  $s_{l_i}, s_{l_i+1}, \dots, s_{r_i}$ . Помогите Сереже ответить на все запросы.

### Формат входных данных

Первая строка содержит последовательность символов без пробелов  $s_1, s_2, \dots, s_n$  ( $1 \leq n \leq 10^6$ ). Каждый символ это либо «(», либо «)». Вторая строка содержит целое число  $m$  ( $1 \leq m \leq 10^5$ ) количество запросов. Каждая из следующих  $m$  строк содержит пару целых чисел. В  $i$ -ой строке записаны числа  $l_i, r_i$ , ( $1 \leq l_i \leq r_i \leq n$ ) — описание  $i$ -го запроса.

### Формат выходных данных

Выведите ответ на каждый запрос в отдельной строке. Ответы выводите в порядке следования запросов во входных данных.

### Пример

стандартный ввод	стандартный вывод
()()()()	0
7	0
1 1	2
2 3	10
1 2	4
1 12	6
8 12	6
5 11	
2 10	

### Замечание

Подпоследовательностью длины  $|x|$  строки  $s = s_1 s_2 \dots s_{|s|}$  (где  $|s|$  — длина строки  $s$ ) называется строка  $x = s_{k_1} s_{k_2} \dots s_{k_{|x|}}$  ( $1 \leq k_1 < k_2 < \dots < k_{|x|} \leq |s|$ ).

Правильной скобочной последовательностью называется скобочная последовательность, которую можно преобразовать в корректное арифметическое выражение путем вставок между ее символами символов «1» и «+». Например, скобочные последовательности «()()», «((())» — правильные (полученные выражения: «(1)+(1)», «((1+1)+1)»), а «()» и «(» — нет.

Для третьего запроса искомая последовательность будет «()».

Для четвертого запроса искомая последовательность будет «()()()».

## Задача D. Поиск максимума

Имя входного файла: `index-max.in`  
Имя выходного файла: `index-max.out`  
Ограничение по времени: 0.5 секунд  
Ограничение по памяти: 256 мегабайт

Реализуйте структуру данных для эффективного вычисления номера максимального из нескольких подряд идущих элементов массива.

### Формат входных данных

В первой строке вводится одно натуральное число  $N$  ( $1 \leq N \leq 100\,000$ ) — количество чисел в массиве.

Во второй строке вводятся  $N$  чисел от 1 до 100 000 — элементы массива.

В третьей строке вводится одно натуральное число  $K$  ( $1 \leq K \leq 30\,000$ ) — количество запросов на вычисление максимума.

В следующих  $K$  строках вводится по два числа — номера левого и правого элементов отрезка массива (считается, что элементы массива нумеруются с единицы).

### Формат выходных данных

Для каждого запроса выведите индекс максимального элемента на указанном отрезке массива. Если максимальных элементов несколько, выведите любой их них.

Числа выводите в одну строку через пробел.

### Пример

<code>index-max.in</code>	<code>index-max.out</code>
5	3
2 2 2 1 5	5
2	
2 3	
2 5	

### Замечание

TL для Python 3.5 секунды

## Задача Е. Взвешивание камней

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

Джек нашел  $N$  камней и упорядочил их в порядке возрастания их массы. Массы всех камней различны. Самый легкий камень получил номер 1, следующий – 2 и так далее, самый тяжелый получил номер  $N$ .

У Джека есть чашечные весы и он решил положить все камни на них в каком-то порядке. Известен порядок, в котором он будет класть камни, и какой камень на какую чашу попадет.

Ваша задача – определить состояние весов после добавления каждого камня. Точные массы камней не известны – даются только их номера.

### Формат входных данных

Первая строка содержит целое число  $N$  ( $1 \leq N \leq 100\,000$ ).

Каждая из следующих  $N$  строк содержит по два целых числа:  $R$  ( $1 \leq R \leq N$ ) и  $S$  ( $1 \leq S \leq 2$ ).  $R$  – номер камня, который будет положен на чашу  $S$ . Все  $R$  будут различны.

### Формат выходных данных

Выведите  $N$  строк – по одной для каждого камня. Если после добавления соответствующего камня чаша 1 тяжелее, выведите «<». Если сторона 2 тяжелее, выведите «>». Если невозможно определить, в каком состоянии будут весы, выведите «?».

### Пример

стандартный ввод	стандартный вывод
5	<
1 2	>
3 1	>
2 1	?
4 2	>
5 1	

## Задача F. 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 входных данных

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 выходных данных

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

### Пример

<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	

### Замечание

TL для Python 10 секунд

## Задача G. Катый ноль

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

Реализуйте эффективную структуру данных, позволяющую изменять элементы массива и вычислять индекс  $k$ -го слева нуля на данном отрезке в массиве.

### Формат входных данных

В первой строке вводится одно натуральное число  $N$  ( $1 \leq N \leq 200\,000$ ) — количество чисел в массиве. Во второй строке вводятся  $N$  чисел от 0 до 100 000 — элементы массива. В третьей строке вводится одно натуральное число  $M$  ( $1 \leq M \leq 200\,000$ ) — количество запросов. Каждая из следующих  $M$  строк представляет собой описание запроса. Сначала вводится одна буква, кодирующая вид запроса (`s` — вычислить индекс  $k$ -го нуля, `u` — обновить значение элемента). Следом за `s` вводится три числа — левый и правый концы отрезка и число  $k$  ( $1 \leq k \leq N$ ). Следом за `u` вводятся два числа — номер элемента и его новое значение.

### Формат выходных данных

Для каждого запроса  $s$  выведите результат. Все числа выводите в одну строку через пробел. Если нужного числа нулей на запрашиваемом отрезке нет, выводите  $-1$  для данного запроса.

### Пример

<code>kthzero.in</code>	<code>kthzero.out</code>
5	4
0 0 3 0 2	
3	
u 1 5	
u 1 0	
s 1 5 3	

### Замечание

TL для Python 8 секунд

## Задача Н. Противник слаб

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 5 секунд  
Ограничение по памяти: 256 мегабайт

Римляне снова наступают. На этот раз их гораздо больше чем персов, но Шапур готов победить их. Он говорит: «Лев никогда не испугается сотни овец».

Не смотря на это, Шапур должен найти слабость римской армии чтобы победить ее. Как вы помните, Шапур — математик, поэтому он определяет насколько слаба армии как число — степень слабости.

Шапур считает, что степень слабости армии равна количеству таких троек  $i, j, k$ , что  $i < j < k$  и  $a_i > a_j > a_k$ , где  $a_x$  — сила человека, стоящего в строю на месте с номером  $x$ . Армия римлян обладает одной особенностью — силы всех людей в ней различны.

Помогите Шапуру узнать, насколько слаба армия римлян.

### Формат входных данных

В первой строке записано одно целое число  $n$  ( $3 \leq n \leq 10^6$ ) — количество солдат в римской армии. Следующая строка содержит  $n$  целых чисел  $a_i$  ( $1 \leq i \leq n, 1 \leq a_i \leq 10^9$ ) — силы людей в римской армии.

### Формат выходных данных

Выведите одно число — степень слабости римской армии.

### Примеры

стандартный ввод	стандартный вывод
3 3 2 1	1
3 2 3 1	0
4 10 8 3 1	4
4 1 5 4 3	1



## Задача I. Число возрастающих подпоследовательностей

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

Задана последовательность из  $n$  чисел  $a_1, a_2, \dots, a_n$ . Подпоследовательностью длины  $k$  этой последовательности называется набор индексов  $i_1, i_2, \dots, i_k$ , удовлетворяющий неравенствам  $1 \leq i_1 < i_2 < \dots < i_k \leq n$ . Подпоследовательность называется возрастающей, если выполняются неравенства  $a_{i_1} < a_{i_2} < \dots < a_{i_k}$ .

Необходимо найти число возрастающих подпоследовательностей наибольшей длины заданной последовательности  $a_1, \dots, a_n$ . Так как это число может быть достаточно большим, необходимо найти остаток от его деления на  $10^9 + 7$ .

### Формат входных данных

Первая строка входного файла содержит целое число  $n$  ( $1 \leq n \leq 10^5$ ). Вторая строка входного файла содержит  $n$  целых чисел:  $a_1, a_2, \dots, a_n$ . Все  $a_i$  не превосходят  $10^9$  по абсолютной величине.

### Формат выходных данных

В выходной файл выведите ответ на задачу.

### Примеры

стандартный ввод	стандартный вывод
5 1 2 3 4 5	1
6 1 1 2 2 3 3	8

## Задача J. Перестановки

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 1 секунда  
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. Количество чисел оказалось довольно большим, поэтому Вася не может окинуть взглядом все числа. Однако ему надо всё-таки представлять эту последовательность, поэтому он написал программу, которая 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 —  $1 \leq N \leq 100\,000$  — количество чисел, которые выписал Вася и  $1 \leq M \leq 100\,000$  — количество вопросов, которые Вася хочет задать программе. Во второй строке дано  $N$  чисел — последовательность чисел, выписанных Васей. Далее в  $M$  строках находятся описания вопросов. Каждая строка содержит четыре целых числа  $1 \leq x \leq y \leq N$  и  $1 \leq k \leq l \leq N$ .

### Формат выходных данных

Выведите  $M$  строк, каждая должна содержать единственное число — ответ на Васин вопрос.

### Пример

стандартный ввод	стандартный вывод
4 2	1
1 2 3 4	3
1 2 2 3	
1 3 1 3	

## Задача К. Сад пермского периода

Имя входного файла: стандартный ввод  
Имя выходного файла: стандартный вывод  
Ограничение по времени: 4 секунды  
Ограничение по памяти: 256 мегабайт

Оранжерея «Сад пермского периода» представляет собой прямоугольный участок для выращивания растений пермского периода. Оранжерея была разбита дорожками на квадраты. В центре каждого квадрата посажено одно растение. Размер квадрата зависит от корневой системы растения.

За год дорожки заросли травой, что затруднило уход за оранжереей. Чтобы при садовых работах не повредить корневую систему какого-либо растения, по имеющемуся расположению растений необходимо восстановить размеры соответствующих им квадратов.

Введем декартову прямоугольную систему координат, начало которой совмещено с левым нижним углом оранжереи. Ось  $Ox$  направлена вдоль нижней границы участка, ось  $Oy$  – вдоль левой. Изначально дорожки были проложены параллельно осям координат. Единичный отрезок удалось выбрать так, что координаты углов каждого из квадратов оказались целыми.

Требуется написать программу, которая по размеру оранжереи и координатам растений определит размеры соответствующих им квадратов.

### Формат входных данных

В первой строке входного файла записаны три натуральных числа:  $W$  – ширина оранжереи,  $H$  – длина оранжереи и  $N$  – количество посаженных растений. В каждой из следующих  $N$  строк расположены по два числа:  $x_i, y_i$  – координаты  $i$ -го растения ( $0 < x_i < W, 0 < y_i < H$ ). Гарантируется, что соответствующие растениям квадраты имеют целую длину стороны и покрывают всю оранжерею.

### Формат выходных данных

В выходной файл необходимо вывести  $N$  целых чисел – размеры квадратов, соответствующих растениям. Числа требуется вывести в порядке описания растений во входном файле.

### Примеры

стандартный ввод	стандартный вывод
4 6 3 1 1 3 1 2 4	2 2 4
8 8 10 4.5 7.5 5.5 7.5 2 6 4.5 6.5 7 7 5 5 6 2 7 5 2 2 5.5 6.5	1 1 4 1 2 2 4 2 4 1
7 7 1 3.5 3.5	7
8 8 1 4 4	8

### Замечание

Оранжерея во втором примере соответствует следующему рисунку:

