

Задача А. Сумма на отрезке

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

Дан массив из N элементов, нужно научиться находить сумму чисел на отрезке.

Формат входных данных

Первая строка входного файла содержит два целых числа N и K — количество чисел в массиве и количество запросов ($1 \leq N \leq 100\,000$, $0 \leq K \leq 100\,000$). Следующие K строк содержат следующие запросы:

1. `A i x` — присвоить i -му элементу массива значение x ($1 \leq i \leq n$, $0 \leq x \leq 10^9$);
2. `Q l r` — найти сумму чисел в массиве на позициях от l до r ($1 \leq l \leq r \leq n$).

Изначально в массиве живут нули.

Формат выходных данных

На каждый запрос вида `Q l r` нужно вывести единственное число — сумму на отрезке.

Примеры

sum.in	sum.out
5 9	0
A 2 2	2
A 3 1	1
A 4 2	2
Q 1 1	0
Q 2 2	5
Q 3 3	
Q 4 4	
Q 5 5	
Q 1 5	

Замечание

TL для Python 4 секунды

Задача В. 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.

Примеры

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

Замечание

TL для Python 2 секунды

Задача С. Катый ноль

Имя входного файла: `kthzero.in`
Имя выходного файла: `kthzero.out`
Ограничение по времени: 1 секунда
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.

Примеры

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

Замечание

TL для Python 10 секунд

Задача D. Знакочередование

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

Реализуйте структуру данных из n элементов a_1, a_2, \dots, a_n , поддерживающую следующие операции:

- присвоить элементу a_i значение j ;
- найти знакочередующуюся сумму на отрезке от l до r включительно, т. е. $(a_l - a_{l+1} + a_{l+2} - \dots a_r)$.

Формат входных данных

В первой строке входного файла содержится натуральное число n ($1 \leq n \leq 10^5$) — длина массива. Во второй строке записаны начальные значения элементов — неотрицательные целые числа, не превосходящие 10^4 .

В третьей строке находится натуральное число m ($1 \leq m \leq 10^5$) — количество операций. В последующих m строках записаны операции:

- операция первого типа задаётся тремя числами $0 \ i \ j$ ($1 \leq i \leq n, 1 \leq j \leq 10^4$).
- операция второго типа задаётся тремя числами $1 \ l \ r$ ($1 \leq l \leq r \leq n$).

Формат выходных данных

Для каждой операции второго типа выведите на отдельной строке соответствующую знакочередующуюся сумму.

Пример

signchange.in	signchange.out
3	-1
1 2 3	2
5	-1
1 1 2	3
1 1 3	
1 2 3	
0 2 1	
1 1 3	

Замечание

TL для Python 4 секунды

Задача Е. Дерево отрезков с операцией на отрезке

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

Реализуйте эффективную структуру данных для хранения элементов и увеличения нескольких подряд идущих элементов на одно и то же число.

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 0 до 100 000 — элементы массива.

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

Каждая из следующих M строк представляет собой описание запроса. Сначала вводится одна буква, кодирующая вид запроса ('g' — получить текущее значение элемента по его номеру, 'a' — увеличить все элементы на отрезке).

Следом за 'g' вводится одно число — номер элемента.

Следом за 'a' вводится три числа — левый и правый концы отрезка и число *add*, на которое нужно увеличить все элементы данного отрезка массива ($0 \leq add \leq 100\,000$).

Формат выходных данных

Выведите в одну строку через пробел ответы на каждый запрос 'g'.

Примеры

segment-tree.in	segment-tree.out
5	4
2 4 3 5 2	2
5	14
g 2	5
g 5	
a 1 3 10	
g 2	
g 4	