

## Задача А. Жадность

Имя входного файла: *стандартный ввод*  
Имя выходного файла: *стандартный вывод*  
Ограничение по времени: 6 секунд  
Ограничение по памяти: 512 мегабайт

В этой задаче вам предстоит решить хорошо известную задачу о рюкзаке. К сожалению, это будет не NP-трудная версия задачи, а более простая модификация.

Дано  $n$  предметов в фиксированном порядке,  $i$ -й имеет вес  $s_i$  и стоимость  $c_i$ . Также есть  $q$  различных рюкзаков, в  $i$ -й из которых помещаются предметы суммарным весом  $w_i$ . Вы заполняете рюкзак, жадным образом помещая в него предметы по одному (помните, что порядок предметов фиксирован и важен). Это значит, что вы никогда не вынимаете предметы и всегда помещаете их в рюкзак, если возможно, то есть если суммарный вес предметов в рюкзаке после этой операции не превысит его вместимости. Вы всегда пытаетесь поместить в рюкзак каждый из  $n$  предметов по порядку независимо от того, получилось ли поместить в него все предыдущие предметы.

Каждый из рюкзаков нужно заполнить по данному алгоритму и вывести суммарную стоимость предметов, которые в него попали. Все рюкзаки заполняются независимо, то есть каждый рюкзак заполняется всеми предметами независимо от того, были ли эти предметы использованы для других рюкзаков.

### Формат входных данных

В первой строке записано целое число  $n$  — количество предметов ( $1 \leq n \leq 10^4$ ).

Во второй строке записано  $n$  целых чисел  $s_1, s_2, \dots, s_n$  — веса предметов ( $1 \leq s_i \leq 10^{13}$ ).

В третьей строке записано  $n$  целых чисел  $c_1, c_2, \dots, c_n$  — стоимости предметов ( $1 \leq c_i \leq 10^4$ ).

В четвёртой строке записано целое число  $q$  — количество рюкзаков, которые нужно попробовать заполнить ( $1 \leq q \leq 10^6$ ).

В пятой строке записаны  $q$  целых чисел  $w_1, w_2, \dots, w_q$  — вместительности рюкзаков ( $1 \leq w_i \leq 10^{18}$ ).

### Формат выходных данных

Выведите  $q$  целых чисел — суммарную стоимость поместившихся вещей для каждого рюкзака.

### Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
5	7
5 3 2 4 1	3
1 2 3 4 5	15
3	
4 8 100	

## Задача В. К-й максимум

Имя входного файла: *стандартный ввод*  
Имя выходного файла: *стандартный вывод*  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 512 мегабайт

Напишите программу, реализующую структуру данных, позволяющую добавлять и удалять элементы, а также находить  $k$ -й максимум, то есть элемент, который бы оказался на  $k$ -м месте, если бы все элементы выписали в порядке убывания.

### Формат входных данных

Первая строка входных данных содержит натуральное число  $n$  — количество команд ( $2 \leq n \leq 100\,000$ ). Последующие  $n$  строк содержат по одной команде каждая. Команда записывается в виде двух чисел  $c_i$  и  $k_i$  — тип и аргумент команды соответственно ( $|k_i| \leq 10^9$ ). Возможные типы команд таковы:

- +1 (или просто 1): Добавить элемент с ключом  $k_i$ .
- 0: Найти и вывести  $k_i$ -й максимум.
- -1: Удалить элемент с ключом  $k_i$ .

Гарантируется, что в процессе работы в структуре не требуется хранить элементы с равными ключами или удалять несуществующие элементы. Также гарантируется, что при запросе  $k_i$ -го максимума он существует.

### Формат выходных данных

Для каждой команды нулевого типа выведите строку, содержащую одно число —  $k_i$ -й максимум.

### Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
11	7
+1 5	5
+1 3	3
+1 7	10
0 1	7
0 2	3
0 3	
-1 5	
+1 10	
0 1	
0 2	
0 3	

## Задача С. Вперёд!

Имя входного файла:	<i>стандартный ввод</i>
Имя выходного файла:	<i>стандартный вывод</i>
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 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 них звучит так: “Рядовые с  $l_i$  по  $r_i$  — вперёд!”

Перед тем, как Дукар отдал первый приказ, солдаты были пронумерованы слева направо целыми числами от 1 до  $n$ . Услышав приказ “Рядовые с  $l_i$  по  $r_i$  — вперёд!”, солдаты, стоящие на местах с  $l_i$  по  $r_i$  включительно, продвигаются в начало ряда в том же порядке, в котором были.

Например, если в какой-то момент солдаты стоят в порядке 1, 3, 6, 2, 5, 4, то после приказа “Рядовые с 2 по 3 — 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 входных данных указаны числа  $n$  и  $m$  — число солдат и число приказов ( $2 \leq n \leq 100\,000$ ,  $1 \leq m \leq 100\,000$ ). Следующие  $m$  строк содержат приказы в виде двух целых чисел:  $l_i$  и  $r_i$  ( $1 \leq l_i \leq r_i \leq n$ ).

### Формат выходных данных

Выведите  $n$  целых чисел — порядок, в котором будут стоять солдаты после исполнения всех приказов.

### Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
6 3	1 4 5 2 3 6
2 4	
3 5	
2 2	

## Задача D. Художник

Имя входного файла:	<i>стандартный ввод</i>
Имя выходного файла:	<i>стандартный вывод</i>
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 мегабайт

Итальянский художник-абстракционист Ф. Мандарино увлёкся рисованием одномерных чёрно-белых картин. Он пытается найти оптимальное местоположение и количество чёрных участков картины. Для этого он проводит на прямой белые и чёрные отрезки, и после каждой из таких операций хочет знать количество чёрных отрезков на получившейся картине и их суммарную длину.

Изначально прямая — белая. Ваша задача — написать программу, которая после каждой из таких операций выводит интересующие художника данные.

### Формат входных данных

В первой строке входных данных содержится общее количество нарисованных отрезков ( $1 \leq N \leq 100\,000$ ). В последующих  $N$  строках содержится описание операций. Каждая операция описывается строкой вида  $c\ x\ l$ , где  $c$  — цвет отрезка (W для белых отрезков, B для чёрных), а сам отрезок имеет вид  $[x; x+l]$ , причём координаты обоих концов — целые числа, не превосходящие по модулю 500 000. Длина задаётся положительным целым числом.

### Формат выходных данных

После выполнения каждой из операций необходимо вывести в отдельной строке количество чёрных отрезков на картине и их суммарную длину, разделив эти числа одним пробелом.

### Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
7	0 0
W 2 3	1 2
B 2 2	1 4
B 4 2	1 4
B 3 2	2 6
B 7 2	3 5
W 3 1	0 0
W 0 10	

## Задача Е. Переворачивания

Имя входного файла: *стандартный ввод*  
Имя выходного файла: *стандартный вывод*  
Ограничение по времени: 3 секунды  
Ограничение по памяти: 512 мегабайт

Учитель физкультуры школы с углублённым изучением предметов уже давно научился считать суммарный рост всех учеников, находящихся в ряду на позициях от  $l$  до  $r$ . Но дети сыграли с ним злую шутку. В некоторый момент дети на позициях с  $l$  по  $r$  меняются местами. Учитель заметил, что у детей не очень богатая фантазия, поэтому они всегда «переворачивают» этот отрезок, то есть  $l$  меняется с  $r$ ,  $l + 1$  меняется с  $r - 1$  и так далее. Учитель решил не ругать детей за их хулиганство, а всё равно посчитать суммарный рост на всех запланированных отрезках. Помогите ему это сделать.

### Формат входных данных

В первой строке записано два числа  $n$  и  $m$  ( $1 \leq n, m \leq 200\,000$ ) — количество детей в ряду и количество событий, произошедших за всё время. Во второй строке задано  $n$  натуральных чисел — рост каждого школьника в порядке следования в ряду. Рост детей не превосходит  $2 \cdot 10^5$ . Далее в  $m$  строках задано описание событий: три числа  $q, l, r$  в каждой строке ( $0 \leq q \leq 1, 1 \leq l \leq r \leq n$ ). Число  $q$  показывает тип события: 0 показывает необходимость посчитать и вывести суммарный рост школьников на отрезке  $[l, r]$ ; 1 показывает то, что дети на отрезке  $[l, r]$  «перевернули» свой отрезок. Все числа во входных данных целые.

### Формат выходных данных

Для каждого события типа 0 выведите единственное число на отдельной строке — ответ на этот запрос.

### Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
5 6	15
1 2 3 4 5	9
0 1 5	8
0 2 4	7
1 2 4	10
0 1 3	
0 4 5	
0 3 5	

## Задача F. Своппер

Имя входного файла:	<i>стандартный ввод</i>
Имя выходного файла:	<i>стандартный вывод</i>
Ограничение по времени:	2 секунды
Ограничение по памяти:	512 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 Сквив.

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 входных данных заданы один или несколько тестовых случаев. В первой строке каждого тестового случая записаны число  $N$  – длина последовательности и число  $M$  – число операций ( $1 \leq N, M \leq 100\,000$ ). Во второй строке тестового случая 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) или 2  $a_i$   $b_i$  (запрос второго типа). Сумма всех  $N$  и  $M$  по всем входным данным не превосходит 200 000. Входные данные завершаются строкой из двух нулей. Гарантируется, что  $x_i < y_i$ , а  $a_i \leq b_i$ .

### Формат выходных данных

Для каждого тестового случая выведите ответы на запросы второго типа, как показано в примере. Разделяйте ответы на тестовые случаи пустой строкой.

### Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
5 5	Swapper 1:
1 2 3 4 5	10
1 2 5	9
2 2 4	2
1 1 4	
2 1 3	
2 4 4	
0 0	

## Задача G. Нарисуй прямой обход

Имя входного файла: *стандартный ввод*  
Имя выходного файла: *стандартный вывод*  
Ограничение по времени: 2 секунды  
Ограничение по памяти: 32 мегабайта (48 мегабайт для Java)

Из википедии мы знаем:

Двоичное дерево поиска (англ. binary search tree, BST) — это двоичное дерево, для каждой вершины которого выполняются следующие дополнительные условия (свойства дерева поиска):

- Оба поддерева — левое и правое — это также двоичные деревья поиска.
- У всех узлов левого поддерева произвольного узла  $x$  значения ключей меньше, нежели значение ключа самого узла  $x$ .
- В то время как значения ключей у всех узлов правого поддерева (того же узла  $x$ ) больше, нежели значение ключа узла  $x$ .

Рассмотрим следующую процедуру вывода BST:

```
void outTree (Node* v) {  
    if (v == 0) return;  
    cout << v->x << " ";  
    outTree(v->left);  
    outTree(v->right);  
}  
outTree(root);
```

Вам дан вывод непустого дерева, сделанный этой функцией. Известно, что по этому выводу можно однозначно восстановить структуру исходного дерева. Преобразуйте его в красивую двумерную картинку.

Обозначим двумерное изображение дерева с корнем в  $v$  как  $rect(v)$ .  $rect(v)$  — прямоугольник, получаемый вызовом рекурсивной функции. Возьмём  $rect(v.left)$ ,  $rect(v.right)$  и строку  $s$ , задающую  $v.x$  ( $s$  тоже прямоугольник, её высота 1, а длина равна длине десятичного представления числа  $v.x$ ). Нарисуем слева направо  $rect(v.left)$ ,  $s$ ,  $rect(v.right)$  таким образом, что правый верхний угол  $rect(v.left)$  совпадает с левым нижним углом  $s$  и верхний левый угол  $rect(v.right)$  совпадает с правым нижним углом  $s$ . Прямоугольник, ограничивающий полученное изображение, и есть  $rect(v)$ .

### Формат входных данных

В единственной строке задан вывод функции `outTree`. Ключи дерева — целые числа от 0 до  $10^9$ . Все ключи различны.

### Формат выходных данных

Выведите 2D-картинку. Гарантируется, что размер вывода не превосходит 5 мегабайт. Строки не должны оканчиваться пробелом. После каждой строки должен следовать перевод строки.

### Пример

<i>стандартный ввод</i>	<i>стандартный вывод</i>
2 1 400 3 500	2 1 400 3 500