

## Задача А. Декартово дерево

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

Как известно, двоичное дерево поиска — это корневое двоичное дерево, в каждой вершине  $v$  которого записано число  $x_v$ , и для любой вершины  $v$  выполнено следующее условие: в любой вершине  $u$  левого поддерева  $v$  число  $x_u < x_v$ , а в любой вершине  $u$  правого поддерева  $v$  число  $x_u > x_v$ .

Двоичное дерево поиска называется *декартовым деревом*, если в каждой вершине  $v$  записано ещё одно число  $y_v$ , и для любой вершины  $v$  выполнено следующее условие: в любой вершине  $u$  каждого поддерева  $v$  число  $y_u > y_v$ . Другими словами, дерево с числами  $y_v$  является двоичной кучей с минимумом.

Дерево называется декартовым, потому что эти свойства наглядны при рассмотрении точек на плоскости с декартовой системой координат. А именно: если нарисовать вершины с координатами  $(x_v, y_v)$  на плоскости, всё левое поддерево вершины  $v$  расположено слева от  $v$ , всё правое поддерево вершины  $v$  расположено справа от  $v$ , и каждое поддерево  $v$  расположено сверху от  $v$ .

Вам даны пары чисел  $(x_i, y_i)$ , причём все  $x_i$  различны, и все  $y_i$  также различны. Постройте декартово дерево, в вершинах которого записаны эти пары, или выясните, что это невозможно.

### Формат входных данных

В первой строке записано число  $n$  — количество пар ( $1 \leq n \leq 50\,000$ ). Каждая из следующих  $n$  строк содержит одну пару чисел:  $x_i$  и  $y_i$ . Для всех пар  $|x_i|, |y_i| \leq 30\,000$ . Гарантируется, что  $x_i \neq x_j$  и  $y_i \neq y_j$  для всех  $i \neq j$ .

### Формат выходных данных

Если декартово дерево с таким набором пар построить возможно, выведите в первой строке «YES», в противном случае выведите «NO».

В случае ответа «YES» выведите ещё  $n$  строк:  $i$ -я строка должна описывать вершину  $i$ . Вершины пронумерованы числами от 1 до  $n$ , в вершине  $i$  записана пара чисел  $(x_i, y_i)$ .

Описание вершины состоит из трёх чисел: номер предка, номер левого сына и номер правого сына. Если у вершины отсутствует предок или какой-либо из сыновей, то выводите на его месте число 0.

Если подходящих деревьев несколько, выведите любое.

## Пример

| <i>стандартный ввод</i> | <i>стандартный вывод</i> |
|-------------------------|--------------------------|
| 7                       | YES                      |
| 5 4                     | 2 3 6                    |
| 2 2                     | 0 5 1                    |
| 3 9                     | 1 0 7                    |
| 0 5                     | 5 0 0                    |
| 1 3                     | 2 4 0                    |
| 6 6                     | 1 0 0                    |
| 4 11                    | 3 0 0                    |

## Задача В. Вставка ключевых значений

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

Вас наняла на работу компания MacroHard, чтобы вы разработали новую структуру данных для хранения целых ключевых значений.

Эта структура выглядит как массив  $A$  бесконечной длины, ячейки которого нумеруются с единицы. Изначально все ячейки пусты. Единственная операция, которую необходимо поддерживать — это операция  $\text{Insert}(L, K)$ , где  $L$  — положение в массиве, а  $K$  — некоторое положительное целое ключевое значение.

Операция выполняется следующим образом:

- Если ячейка  $A[L]$  пуста, то присвоить  $A[L] := K$ .
- Если ячейка  $A[L]$  непуста, выполнить  $\text{Insert}(L + 1, A[L])$ , а затем присвоить  $A[L] := K$ .

По заданной последовательности из  $N$  целых чисел  $L_1, L_2, \dots, L_N$  вам необходимо вывести содержимое этого массива после выполнения следующей последовательности операций:

$\text{Insert}(L_1, 1)$   
 $\text{Insert}(L_2, 2)$   
 $\dots$   
 $\text{Insert}(L_N, N)$

## Формат входных данных

В первой строке содержатся  $N$  — число операций  $\text{Insert}$  — и  $M$  — максимальный номер позиции, которую можно использовать в операции  $\text{Insert}$  ( $1 \leq N \leq 131\,072$ ,  $1 \leq M \leq 131\,072$ ).

В следующей строке даны  $N$  целых чисел  $L_i$ , которые описывают операции  $\text{Insert}$  ( $1 \leq L_i \leq M$ ).

## Формат выходных данных

Выведите содержимое массива после выполнения данной последовательности операций  $\text{Insert}$ . В первой строке выведите  $W$  — номер последней несвободной позиции в массиве. Далее выведите  $W$  целых чисел:  $A[1], A[2], \dots, A[W]$ . Для пустых ячеек выводите нули.

## Пример

| <i>стандартный ввод</i> | <i>стандартный вывод</i> |
|-------------------------|--------------------------|
| 5 4                     | 6                        |
| 3 3 4 1 3               | 4 0 5 2 3 1              |

## Задача С. 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 D. 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 Е. Менеджер памяти

Имя входного файла: *стандартный ввод*  
Имя выходного файла: *стандартный вывод*  
Ограничение по времени: 10 секунд  
Ограничение по памяти: 512 мегабайт

Одно из главных нововведений новейшей операционной системы Indows 7 — новый менеджер памяти. Он работает с массивом длины  $N$  и позволяет выполнять три самые современные операции:

- $\text{copy}(a, b, \ell)$  — скопировать отрезок  $[a, a + \ell - 1]$  в отрезок  $[b, b + \ell - 1]$ ;
- $\text{sum}(\ell, r)$  — посчитать сумму элементов массива на отрезке  $[\ell, r]$ ;
- $\text{print}(\ell, r)$  — напечатать элементы с индексами от  $\ell$  до  $r$  включительно.

Вы разрабатываете свою операционную систему и, безусловно, не можете обойтись без инновационных технологий. Вам необходимо реализовать точно такой же менеджер памяти.

### Формат входных данных

Первая строка содержит целое число  $N$  ( $1 \leq N \leq 1\,000\,000$ ) — размер массива, с которым будет работать менеджер памяти.

Во второй строке содержатся четыре целых числа  $X_1, A, B, M$  ( $1 \leq X_1, A, B, M \leq 10^9 + 10$ ). С помощью них можно сгенерировать исходный массив чисел  $X_1, X_2, \dots, X_N$  по следующему правилу:  $X_{i+1} = (A \cdot X_i + B) \bmod M$ .

Следующая строка содержит целое число  $K$  ( $1 \leq K \leq 200\,000$ ) — количество запросов, которые необходимо выполнить менеджеру памяти.

Далее в  $K$  строках содержится описание запросов. Запросы заданы в следующем формате:

- $\text{cpy } a \ b \ \ell$  — для операции  $\text{copy}$ ;
- $\text{sum } \ell \ r$  — для операции  $\text{sum}$  ( $\ell \leq r$ );
- $\text{out } \ell \ r$  — для операции  $\text{print}$  ( $\ell \leq r$ ).

Гарантируется, что суммарная длина запросов  $\text{out}$  не превышает 3000. Также гарантируется, что все запросы корректны.

## Формат выходных данных

Для каждого запроса  $\text{sum}$  или  $\text{out}$  выведите результат в отдельной строке.

### Пример

| <i>стандартный ввод</i> | <i>стандартный вывод</i> |
|-------------------------|--------------------------|
| 6                       | 1 2 6 1 2 6              |
| 1 4 5 7                 | 18                       |
| 8                       | 1 2                      |
| out 1 6                 | 3                        |
| sum 1 6                 | 1 1 2 1 2 6              |
| cpy 1 3 2               | 13                       |
| out 3 4                 |                          |
| sum 3 4                 |                          |
| cpy 1 2 4               |                          |
| out 1 6                 |                          |
| sum 1 6                 |                          |

## Задача F. Вперёд!

Имя входного файла: *стандартный ввод*  
Имя выходного файла: *стандартный вывод*  
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  $\ell_i$  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  $\ell_i$  по  $r_i$  — вперёд!", солдаты, стоящие на местах с  $\ell_i$  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:  $\ell_i$  и  $r_i$  ( $1 \leq \ell_i \leq r_i \leq n$ ).

### 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<br>2 4<br>3 5<br>2 2 | 1 4 5 2 3 6              |

## Задача G. Художник

Имя входного файла: *стандартный ввод*  
Имя выходного файла: *стандартный вывод*  
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  $s\ x\ \ell$ , где  $s$  — цвет отрезка (W для белых отрезков, B для чёрных), а сам отрезок имеет вид  $[x; x + \ell]$ , 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<br>W 2 3<br>B 2 2<br>B 4 2<br>B 3 2<br>B 7 2<br>W 3 1<br>W 0 10 | 0 0<br>1 2<br>1 4<br>1 4<br>2 6<br>3 5<br>0 0 |

## Задача Н. 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  $\ell$  до  $r$ . Но дети сыграли с ним злую шутку. В некоторый момент дети на позициях с  $\ell$  по  $r$  меняются местами. Учитель заметил, что у детей не очень богатая фантазия, поэтому они всегда «переворачивают» этот отрезок, то есть  $\ell$  меняется с  $r$ ,  $\ell+1$  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  $q, \ell, r$  в каждой строке ( $0 \leq q \leq 1, 1 \leq \ell \leq r \leq n$ ). Число  $q$  показывает тип события: 0 показывает необходимость посчитать и вывести суммарный рост школьников на отрезке  $[\ell, r]$ ; 1 показывает то, что дети на отрезке  $[\ell, r]$  «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 I. И снова сумма...

Имя входного файла: *стандартный ввод*  
Имя выходного файла: *стандартный вывод*  
Ограничение по времени: 5 секунд  
Ограничение по памяти: 512 мегабайт

Реализуйте структуру данных, которая поддерживает множество  $S$  целых чисел, к которому поступают следующие запросы:

- `add ( $i$ )` — добавить в множество  $S$  число  $i$  (если оно там уже есть, то множество не меняется);
- `sum ( $\ell, r$ )` — вывести сумму всех элементов  $x$  из  $S$ , которые удовлетворяют неравенству  $\ell \leq x \leq r$ .

### Формат входных данных

Исходно множество  $S$  пусто. Первая строка входных данных содержит целое число  $n$  — количество операций ( $1 \leq n \leq 300\,000$ ). Следующие  $n$  строк содержат операции. Каждая операция выглядит либо как «+  $i$ », либо как «?  $\ell r$ ».

Операция «?  $\ell r$ » задаёт запрос «sum ( $\ell, r$ )».

Если операция «+  $i$ » находится в начале входных данных или следует непосредственно после другой операции «+», то она задаёт запрос «add ( $i$ )». Если же она следует непосредственно после запроса «?», и результат этого запроса был  $y$ , то выполняется операция «add ( $v$ )», где  $v = (i + y) \bmod 10^9$ .

Во всех операциях параметры лежат в интервале от 0 до  $10^9$  включительно.

### Формат выходных данных

Для каждого запроса выведите одно число — ответ на запрос.

### Пример

| <i>стандартный ввод</i> | <i>стандартный вывод</i> |
|-------------------------|--------------------------|
| 6                       | 3                        |
| + 1                     | 7                        |
| + 3                     |                          |
| + 3                     |                          |
| ? 2 4                   |                          |
| + 1                     |                          |
| ? 2 4                   |                          |

## Задача J. Своппер

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

Современные компьютеры закидываются 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 обшчёт 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                     |                          |