1A. RMQ

1 секунда, 256 мегабайт

Реализуйте структуру данных, которая на данном массиве из N целых чисел позволяет узнать максимальное значение на этом массиве и индекс элемента, на котором достигается это максимальное значение.

Входные данные
В первой строке вводится натуральное число N ($1 \le N \le 10^5$) – количество элементов в массиве. В следующей строке содержатся N целых чисел, не превосходящих по модулю 10^9 – элементы массиваб гарантируется, что в массиве нет одинаковых элементов. Далее идет число K ($0 \le K \le 10^5$) – количество запросов к структуре данных. Каждая из следующих K строк содержит два целых числа l и r ($1 \le l \le r \le N$) – левую и правую границы отрезка в массиве для данного запроса.

Выходные данные
Для каждого из запросов выведите два числа: наибольшее значение среди элементов массива на отрезке от l до r и индекс одного из элементов массива, принадлежащий отрезку от l до r, на котором достигается этот максимум.

входные данные
5 7 3 1 6 4 3 1 5 2 4 3 3
выходные данные
7 1 6 4 1 3

1B. Количество максимумов на отрезке

1 секунда, 256 мегабайт

Реализуйте структуру данных для эффективного вычисления значения максимального из нескольких подряд идущих элементов массива, а также количества элементов, равных максимальному на данном отрезке.

Входные данные
В первой строке вводится одно натуральное число N ($1 \le N \le 100\,000$) — 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 ($1 \le K \le 30\,000$) — 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).

Выходные данные
Для каждого запроса выведите в отдельной строке через пробел значение максимального элемента на указанном отрезке массива и количество максимальных элементов на этом отрезке.

входные данные
5 2 2 2 1 5 2 2 3 2 5
выходные данные
2 2 5 1

1C. Нолики

1 секунда, 256 мегабайт

Дедус любит давть 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.

Входные данные
В первой строке входного файла записано число N ($1 \le N \le 500\,000$) – 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 ($1 \le M \le 50\,000$) – количество запросов. Каждая из следующих M строк содержит описания запросов:

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

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

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

входные данные
5 328 0 0 0 0 5 QUERY 1 3 UPDATE 2 832 QUERY 3 3 QUERY 2 3 UPDATE 2 0
выходные данные
2 1 1

1D. Катый ноль

2 секунды, 256 мегабайт

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

Входные данные
В первой строке вводится одно натуральное число N ($1 \le N \le 200\,000$) — 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 ($1 \le M \le 200\,000$) — 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 ($1 \le k \le N$). Следом за `u` вводятся два числа — номер элемента и его новое значение.

Выходные данные
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.

входные данные
5 0 0 3 0 2 3 u 1 5 u 1 0 s 1 5 3

выходные данные
4

TL для Python 8 секунд

1E. Ближайшее большее число справа

2 секунды, 256 мегабайт

Дан массив a из n чисел. Нужно обрабатывать запросы:

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

Входные данные

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.

Выходные данные

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

входные данные
4 5 1 2 3 4 1 1 1 1 1 3 1 1 5 0 2 3 1 1 3
выходные данные
1 3 -1 2

1F. Число возрастающих подпоследовательностей

1 секунда, 256 мегабайт

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

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

Входные данные

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

Выходные данные

В выходной файл выведите ответ на задачу.

входные данные
5 1 2 3 4 5
выходные данные
1

входные данные
6 1 1 2 2 3 3
выходные данные
8

1G. Противник слаб

5 секунд, 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.

Помогите Шапуру узнать, насколько слаба армия римлян.

Входные данные

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.

Выходные данные

Выведите одно число — степень слабости римской армии.

входные данные
3 3 2 1
выходные данные
1

входные данные
3 2 3 1
выходные данные
0

входные данные
4 10 8 3 1
выходные данные
4

входные данные
4 1 5 4 3
выходные данные
1

1H. Сережа и скобочки

1 секунда, 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 $s_{l_i}, s_{l_{i+1}}, \ldots, s_{r_i}$. Помогите Сереже ответить на все запросы.

Входные данные

Первая строка содержит последовательность символов без пробелов $s_1, s_2, \ldots, s_n (1 \leq n \leq 10^6)$. 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.

Выходные данные

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

входные данные
())(())(())(7 1 1 2 3 1 2 1 12 8 12 5 11 2 10

выходные данные
0 0 2 10 4 6 6

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

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 «()(())(())».

1I. Марио и трубы

4.0 с, 256 MB

Марио собирается проходить уровень, состоящий из N последовательно расположенных труб, высота i-й трубы — a_i. Он еще не знает, где он будет располагаться изначально, и куда ему надо добраться, поэтому хочет рассмотреть несколько вариантов.

Находясь на трубе, Марио может переместиться только на соседние трубы слева и справа (если они существуют). Спускаться он может с любой высоты, также он может перемещаться между одинаковыми трубами. Подниматься Марио может только на трубу, высота которой больше высоты текущей на 1. Более формально, Марио может переместиться с трубы i на трубу j, если $|i - j| = 1$ и $a_j - a_i \le 1$.

Однако злой динозавр Боузер хочет помешать Марио пройти уровень, для чего иногда увеличивает высоту нескольких подряд идущих труб на одно число k. Теперь Марио не может понять, удастся ли ему пройти уровень и поэтому просит вас обрабатывать два типа запросов — Боузер изменяет высоту некоторых труб, и Марио пытается пройти от одной трубы до другой.

Входные данные

В первой строке заданы два целых числа N и M — число труб и число запросов соответственно ($2 \le N \le 3 \cdot 10^5, 1 \le M \le 10^6$).

Следующая строка содержит N целых чисел a_i — высоты труб на уровне ($1 \le a_i \le 10^9$).

Далее идут M строк, содержащие описание запросов. Каждая строка имеет вид:

- `1 x y` — может ли Марио пройти от трубы с номером x до трубы с номером y ($1 \le x, y \le N$). Гарантируется, что номера x и y не совпадают.
- `2 l r d` — Боузер увеличивает высоты труб с l-й до r-й на величину d ($1 \le l \le r \le N$, $-10^9 \le d \le 10^9$).

Выходные данные

Для каждого запроса запроса первого типа нужно на отдельной строке вывести «YES», если Марио может дойти от одной трубы до другой и «NO» в противном случае (без кавычек).

входные данные
5 7 1 2 3 4 5 1 5 1 2 2 4 3 1 5 4 1 1 3 2 2 3 3 1 2 4 1 1 3

выходные данные
YES NO NO YES NO

2A. Максимум на подотрезках с добавлением на отрезке

0.5 секунд, 256 мегабайт

Реализуйте эффективную структуру данных для хранения массива и выполнения следующих операций: увеличение всех элементов данного интервала на одно и то же число; поиск максимума на интервале.

Входные данные

В первой строке вводится одно натуральное число $N (1 \le N \le 100000)$ — количество чисел в массиве.

Во второй строке вводятся N чисел от 0 до 100000 — элементы массива.

В третьей строке вводится одно натуральное число $M (1 \le M \le 30000)$ — количество запросов.

Каждая из следующих M строк представляет собой описание запроса. Сначала вводится одна буква, кодирующая вид запроса (m — найти максимум, a — увеличить все элементы на отрезке).

Следом за m вводятся два числа — левая и правая граница интервала.

Следом за a вводятся три числа — левый и правый концы отрезка и число add, на которое нужно увеличить все элементы данного отрезка массива $(0 \le add \le 100000)$.

Выходные данные

Выведите в одну строку через пробел ответы на каждый запрос m.

входные данные
5 2 4 3 1 5 5 m 1 3 a 2 4 100 m 1 3 a 5 5 10 m 1 5

выходные данные
4 104 104

2B. XOR на отрезке

3 секунды, 512 мегабайт

Вам задан массив a, состоящий из n целых чисел $a_1, a_2, \dots, a_n$. С этим массивом разрешается выполнять две операции:

1. Вычислить сумму текущих элементов массива на отрезке $[l, r]$, то есть посчитать значение $a_l + a_{l+1} + \dots + a_r$
2. Применить операцию хог с заданным числом x к каждому элементу массива на отрезке $[l, r]$, то есть выполнить $a_l = a_l \oplus x, a_{l+1} = a_{l+1} \oplus x, \dots, a_r = a_r \oplus x$. Эта операция изменяет ровно $r - l + 1$ элементов массива.

Выражение $x \oplus y$ означает применение побитовой операции хог к числам x и y.

Вам задан список из m операций указанного вида. От Вас требуется выполнить все заданные операции, для каждого запроса суммы требуется вывести полученный результат.

Входные данные

Входные данные В первой строке задано целое число n $(1 \le n \le 10^5)$ - размер массива. Во второй строке через пробел заданы целые числа $a_1, a_2, \ldots, a_n$ $(0 \le a_i \le 10^6)$ — исходный массив.

В третьей строке задано целое число m ($1 \le m \le 10^5$) - количество операций с массивом. В i -ой из следующих m строк сперва записано целое число t_i $(1 \le t_i \le 2)$ — тип i -го запроса. Если $t_i = 1$, то это запрос суммы, если $t_i = 2$, то это запрос на изменение элементов массива. Если i -ая операция типа 1, то далее следуют два целых числа l_i, r_i $(1 \le l_i \le r_i \le n)$. Если i -ая операция типа 2, то далее следуют три целых числа l_i, r_i, x_i $(1 \le l_i \le r_i \le n, 1 \le x_i \le 10^6)$. Числа в строках разделены одиночными пробелами.

Выходные данные

Для каждого запроса типа 1 в отдельной строке выведите сумму чисел на требуемом отрезке. Ответы на запросы выводите в том порядке, в котором они заданы во входных данных.

входные данные
5 4 10 3 13 7 8 1 2 4 2 1 3 3 1 2 4 1 3 3 2 2 5 5 1 1 5 2 1 2 10 1 2 3
выходные данные
26 22 0 34 11

входные данные
6 4 7 4 0 7 3 5 2 2 3 8 1 1 5 2 3 5 1 2 4 5 6 1 2 3
выходные данные
38 28

2C. Объединение прямоугольников

1 секунда, 256 мегабайт

Дано N прямоугольников со сторонами, параллельными осям координат и вершинами в целочисленных точках. Найдите площадь их объединения.

Входные данные

В первой строке дано число $0 \le N < 10^5$ -количество прямоугольников. В следующих N строках даны описания прямоугольников. Каждое описание прямоугольника — это 4 числа через пробел: $\langle x_1, y_1, x_2, y_2 \rangle$. Левый нижний угол прямоугольника имеет координаты $\langle x_1, x_2 \rangle$, правый верхний угол имеет координаты $\langle x_2, y_2 \rangle$.

$-10^9 \le x_1 \le x_2 \le 10^9$;

$-10^9 \le y_1 \le y_2 \le 10^9$

Выходные данные

Выведите одно число — площадь объединения этих прямоугольников

входные данные
2 0 0 2 2 1 3 2 4
выходные данные
5

входные данные
0
выходные данные
0

входные данные
3 1 1 3 5 5 2 7 4 2 4 6 7
выходные данные
23

2D. Простое задание

2 секунды, 512 мегабайт

Это задание очень простое. Вам дана строка S длины n и q запросов, каждый запрос имеет формат i j k, что означает: отсортировать подстроку, состоящую из символов от i до j, в неубывающем порядке, если $k = 1$ или в невозрастающем порядке, если $k = 0$.

Выведите итоговую строку после выполнения запросов.

Входные данные

В первой строке записано два целых числа n, q ($1 \le n \le 10^5$, $0 \le q \le 50000$), длина строки и количество запросов, соответственно.

В следующей строке идёт сама строка S. Она состоит только из строчных английских букв.

В каждой из следующих q строк записано по три целых числа i, j, k ($1 \le i \le j \le n$, $0 \le k \le 1$), обозначающих запрос.

Выходные данные

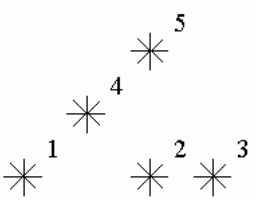
Выведите строку S после выполнения всех запросов.

входные данные
10 10 ittmcsvmoa 6 7 0 2 4 0 4 10 1 1 2 0 2 9 1 5 6 1 7 9 0 1 2 0 2 6 0 3 9 1
выходные данные
tmacimostv

2E. Звезды

1 секунда, 256 мегабайт

Астрономы часто изучают звездные карты, на которых звезды обозначены точками на плоскости, и каждая звезда задана своими декартовыми координатами. Назовем уровнем звезды количество звезд, которые расположены не выше и не правее данной звезды. Астрономы хотят знать распределение уровней звезд.



В качестве примера рассмотрим карту, показанную на рисунке выше. Уровень звезды 5 равен трем (уровень сформирован звездами с номерами $1, 2$ и 4). Уровни звезд с номерами 2 и 4 равны одному. На данной карте есть единственная звезда с уровнем, равным нулю, две звезды с уровнем, равным одному, одна звезда с уровнем два и одна звезда с уровнем три.

От вас требуется написать программу, которая посчитает количество звезд каждого уровня на заданной карте.

Входные данные
Первая строка содержит число N — количество звезд на карте $(1 \le N \le 150000)$.

Следующие N строк описывают координаты звезд. В каждой строке находятся два целых числа, разделенные пробелом — X_i и Y_i $(0 \le X_i, Y_i \le 200000)$. В каждой точке плоскости находится не более одной звезды. Звезды перечислены в порядке увеличения координаты Y. Звезды с равными Y-координатами перечислены в порядке увеличения координаты X.

Выходные данные
Ваша программа должна вывести N строк. В i-й строке должно быть записано одно целое число — количество звезд уровня $i-1$.

входные данные
5 1 1 5 1 7 1 3 3 5 5
выходные данные
1 2 1 1 0

2F. Окна

1 секунда, 256 мегабайт

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

Входные данные
В первой строке входного файла записано число окон n ($1 \le n \le 50\,000$).

Следующие n строк содержат координаты окон $x_{i,1}, y_{i,1}, x_{i,2}, y_{i,2}$, где $(x_{i,1}, y_{i,1})$ — координаты левого верхнего угла i-го окна, а $(x_{i,2}, y_{i,2})$ — правого нижнего (на экране компьютера y растет сверху вниз, а x — слева направо). Все координаты — целые числа, по модулю не превосходящие 10^6.

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

входные данные
2 0 0 3 3 1 1 4 4
выходные данные
2 1 3

входные данные
1 0 0 1 1
выходные данные
1 0 1

входные данные
4 0 0 1 1 0 1 1 2 1 0 2 1 1 1 2 2
выходные данные
4 1 1

2G. Петя и массив

2 секунды, 256 мегабайт

У Пети есть массив a, состоящий из n целых чисел. Недавно он изучил префикс суммы и теперь умеет быстро считать сумму элементов на любом подотрезке своего массива. Под подотрезком понимается непустая последовательность подряд идущих элементов массива.

Теперь ему стало интересно, сколько в его массиве подотрезков с суммой элементов меньше t. Помогите Пете и определите это количество.

Формально, нужно найти количество пар l, r ($l \le r$), что $a_l + a_{l+1} + \cdots + a_{r-1} + a_r < t$.

Входные данные
В первой строке следуют два целых числа n и t ($1 \le n \le 200\,000, |t| \le 2 \cdot 10^{14}$).

Во второй строке следует последовательность целых чисел $a_1, a_2, \ldots, a_n$ ($|a_i| \le 10^9$) — описание массива Пети. Обратите внимание, что элементы могут быть и отрицательными, и нулевыми, и положительными.

Выходные данные
Выведите количество подотрезков в массиве Пети с суммой элементов меньше t.

входные данные
5 4 5 -1 3 4 -1
выходные данные
5

входные данные
3 0 -1 2 -3
выходные данные
4

входные данные
4 -1 -2 1 -2 3
выходные данные
3

В первом примере у следующих отрезки сумма меньше 4:

- $[2, 2]$, сумма на отрезке равна -1
- $[2, 3]$, сумма на отрезке равна 2
- $[3, 3]$, сумма на отрезке равна 3
- $[4, 5]$, сумма на отрезке равна 3
- $[5, 5]$, сумма на отрезке равна -1

2H. Продажа картин

4 секунды, 256 мегабайт

Лука — торговец картинами. У него есть N клиентов, каждому из которых он продает произведения искусства. Каждый клиент может купить либо цветные картины, либо черно-белые, но не те и другие вместе. При этом клиент под номером i готов купить не более a_i цветных картин и не более b_i черно-белых картин. При этом каждый клиент хочет купить хотя бы одну картину.

У Луки практически неограниченный запас картин, поэтому запросы клиентов не являются для него проблемой. Однако, Лука не любит продавать черно-белые картины, и если окажется, что меньше, чем C людей купили цветные картины, он очень огорчится.

В силу нестабильной экономической ситуации в стране клиенты постоянно изменяют свои запросы, иными словами количество цветных и черно-белых картин, которые они готовы купить. Из-за этого Лука постоянно задается вопросом: «Сколько у меня есть вариантов, как продать клиентам картины, чтобы хотя бы C человек купили цветные картины?». Помогите Луке и защитите его от излишнего беспокойства.

Входные данные

Первая строка содержит два целых числа N и C ($1 \le N \le 10^5, 1 \le C \le 20$).

Вторая строка содержит N целых чисел a_i ($1 \le a_i \le 10^9$).

Третья строка содержит N целых чисел b_i ($1 \le b_i \le 10^9$).

Четвертая строка содержит одно целое число Q ($1 \le Q \le 10^5$) — количество изменений требований клиентов.

Каждая из следующих Q строк содержит три числа: номер клиента, меняющего требования P ($1 \le P \le N$), новое максимальное количество цветных картин, которое он готов купить A_p ($1 \le A_p \le 10^9$) и новое максимальное количество черно-белых картин, которое он готов купить B_p ($1 \le B_p \le 10^9$).

Выходные данные

Выведите Q строк, где в q-й строке записано единственное число — количество вариантов продать картины клиентам, чтобы хотя бы C человек купили цветные картины, по модулю 10007 после первых q изменений требований.

Система оценки

Подзадача	Баллы	Дополнительные ограничения	Оценка	Н
0	0	Тесты из условия	подзадача	
1	30	$1 \le N, Q \le 1000$	подзадача	
2	70	Дополнительных ограничений нет	подзадача	

входные данные
2 2 1 1 1 1 1 1 1 1
выходные данные
1

входные данные
2 2 1 2 2 3 2 1 2 2 2 2 2
выходные данные
4 4

входные данные
4 2 1 2 3 4 1 2 3 4 1 4 1 1
выходные данные
66

2I. Робот-пылесос

1 s., 512 MB

Рассмотрим координатную плоскость, которую планируется очищать с использованием робота пылесоса. Робот-пылесос представляет собой квадрат размером $k \times k$ со сторонами, параллельными осям координат. Изначально левый нижний угол робота находится в точке $(0, 0)$, а правый верхний, соответственно — в точке (k, k).

Вам дана последовательность из n перемещений робота по плоскости, i-е перемещение характеризуется направлением d_i, принимающим значения 'N' (вверх, увеличение координаты Y), 'S' (вниз, уменьшение координаты Y), 'W' (влево, уменьшение координаты X) или 'E' (вправо, увеличение координаты X), и целым числом a_i — расстоянием, на которое робот перемещается.

Робот в каждый момент времени убирает всю площадь под собой. Иными словами, точка считается убранной тогда и только тогда, когда она в какой-то момент времени принадлежала квадрату размера $k \times k$, на котором находился робот.

По заданным перемещениям робота посчитайте суммарную площадь всей убранной поверхности.

Входные данные

В первой строке ввода через пробел даны два целых числа: размер робота k и количество команд n ($1 \le k \le 10^4$; $1 \le n \le 10^5$).

В i-й из следующих n строк через пробел даны направление i-го перемещения d_i и его расстояние a_i (d_i — буква 'N', 'S', 'W' или 'E'; $1 \le a_i \le 10^9$).

Выходные данные

Выведите суммарную площадь убранной роботом поверхности.

входные данные
1 5 E 2 N 2 W 4 S 4 E 4
выходные данные
17

входные данные
3 4 W 2 N 1 W 1 N 2
выходные данные
27

3A. Разреженные таблицы

1 секунда, 256 мегабайт

Дан массив из n чисел. Требуется написать программу, которая будет отвечать на запросы следующего вида: найти минимум на отрезке между u и v включительно.

Входные данные

В первой строке входного файла даны три натуральных числа n, m ($1 \leqslant n \leqslant 10^5$, $1 \leqslant m \leqslant 10^7$) и a_1 ($0 \leqslant a_1 < 16\,714\,589$) — количество элементов в массиве, количество запросов и первый элемент массива соответственно. Вторая строка содержит два натуральных числа u_1 и v_1 ($1 \leqslant u_1, v_1 \leqslant n$) — первый запрос.

Элементы $a_2, a_3, \ldots, a_n$ задаются следующей формулой:

$$a_{i+1} = (23 \cdot a_i + 21563) \bmod 16714589.$$

Например, при $n = 10$, $a_1 = 12345$ получается следующий массив: a = (12345, 305498, 7048017, 11694653, 1565158, 2591019, 9471233, 570265, 13137658, 1325095).

Запросы генерируются следующим образом:

$$u_{i+1} = \big((17 \cdot u_i + 751 + ans_i + 2i) \bmod n\big) + 1,$$
$$v_{i+1} = \big((13 \cdot v_i + 593 + ans_i + 5i) \bmod n\big) + 1,$$

где ans_i — ответ на запрос номер i.
Обратите внимание, что u_i может быть больше, чем v_i.

Выходные данные

В выходной файл выведите u_m, v_m и ans_m (последний запрос и ответ на него).

входные данные
10 8 12345 3 9
выходные данные
5 3 1565158

3B. Звёзды

2 секунды, 256 мегабайт

Вася любит наблюдать за звёздами. Но следить за всем небом сразу ему тяжело. Поэтому он наблюдает только за частью пространства, ограниченной кубом размером $n \times n \times n$. Этот куб поделен на маленькие кубики размером $1 \times 1 \times 1$. Во время его наблюдений могут происходить следующие события:

1. В каком-то кубике появляются или исчезают несколько звёзд.

2. К нему может заглянуть его друг Петя и поинтересоваться, сколько видно звезд в части пространства, состоящей из нескольких кубиков.

Входные данные
Первая строка входного файла содержит натуральное число $1 \le n \le 128$. Координаты кубиков — целые числа от 0 до $n - 1$. Далее следуют записи о происходивших событиях по одной в строке. В начале строки записано число m. Если m равно:

1. Тогда за ним следуют 4 числа — x, y, z ($0 \le x, y, z < N$) и k ($-20000 \le k \le 20000$) — координаты кубика и величина, на которую в нем изменилось количество видимых звёзд;
2. Тогда за ним следуют 6 чисел — x_1, y_1, z_1, x_2, y_2, z_2 ($0 \le x_1 \le x_2 < N$, $0 \le y_1 \le y_2 < N$, $0 \le z_1 \le z_2 < N$), которые означают, что Петя попросил подсчитать количество звезд в кубиках (x, y, z) из области: $x_1 \le x \le x_2$, $y_1 \le y \le y_2$, $z_1 \le z \le z_2$;
3. Это означает, что Васе надоело наблюдать за звёздами и отвечать на вопросы Пети. Эта запись встречается во входном файле только один раз и будет последней.

Количество записей во входном файле не больше $100\,002$.

Выходные данные
Для каждого Петиного вопроса выведите искомое количество звёзд.

входные данные
2 2 1 1 1 1 1 1 1 0 0 0 1 1 0 1 0 3 2 0 0 0 0 0 0 2 0 0 0 0 1 0 1 0 1 0 -2 2 0 0 0 1 1 1 3
выходные данные
0 1 4 2

3C. Друзья и последовательности

2 секунды, 512 мегабайт

Майк и !Майк соперничают еще со школьных лет, они противоположны во всем что делают, кроме программирования. Сегодня у них возникла проблема, которую сами друзья сами решить не могут, но вместе с вами — кто знает?

Каждый из них знает две последовательности n чисел a и b. По запросу в виде пары целых чисел (l, r) Майк может сразу сообщить значение $\max\limits_{i=l}^{r} a_i$, а !Майк — значение $\min\limits_{i=l}^{r} b_i$.

Предположим, что робот задает им каждый из возможных различных запросов в виде пары целых чисел (l, r) $(1 \le l \le r \le n)$ (то есть он сделает ровно $n(n+1)/2$ запросов) и считает, сколько раз их ответы на один и тот же запрос совпадают, то есть для скольких пар выполняется $\max\limits_{i=l}^{r} a_i = \min\limits_{i=l}^{r} b_i$.

Сколько случаев совпадения посчитает робот?

Входные данные
В первой строке содержится единственное целое число n $(1 \le n \le 200\,000)$.

Во второй строке содержатся n целых чисел $a_1, a_2, ..., a_n$ $(-10^9 \le a_i \le 10^9)$ — элементы последовательности a.

В третьей строке содержатся n целых чисел $b_1, b_2, ..., b_n$ $(-10^9 \le b_i \le 10^9)$ — элементы последовательности b.

Выходные данные
Выведите одно целое число — количество совпадений ответов, которые посчитает робот, то есть для скольких пар выполняется $\max\limits_{i=l}^{r} a_i = \min\limits_{i=l}^{r} b_i$.

входные данные
6 1 2 3 2 1 4 6 7 1 2 3 2
выходные данные
2

входные данные
3 3 3 3 1 1 1
выходные данные
0

В первом примере входных данных всего два совпадения:

1. $l = 4, r = 4$ since $max\{2\} = min\{2\}$.

2. $l = 4, r = 5$ since $max\{2, 1\} = min\{2, 3\}$.

Во втором примере входных данных совпадений не произойдет, так как ответ Майка на любой запрос всегда 3, в то время как !Майк всегда будет отвечать 1.

3D. Ор выше гор

1 секунда, 512 мегабайт

Рик и Морти очень любят ходить на горный хребет Высокоорный для того, чтобы поорать — эхо там просто невероятное. Не так давно они нашли интересное акустическое свойство этого хребта: если Рик и Морти начнут одновременно орать с разных гор, то их ор будет слышен между этими горами вплоть до высоты, равной побитовому ИЛИ высот гор, на которые они взошли, и всех гор между ними.

Побитовое ИЛИ — это бинарная операция, которая определяется следующим образом. Рассмотрим записи чисел x и y в двоичной системе счисления (возможно с ведущими нулями) $x = x_k... x_1x_0$ и $y = y_k... y_1y_0$. Тогда $z = x \mid y$ определяется следующим образом: $z = z_k... z_1z_0$, где $z_i = 1$, если $x_i = 1$ или $y_i = 1$, иначе $z_i = 0$. Иными словами, нули в побитовом ИЛИ чисел находятся только в тех разрядах, в которых у обоих чисел находятся нули. Например, побитовое ИЛИ чисел $10 = 1010_2$ и $9 = 1001_2$ равняется $11 = 1011_2$. В языках программирования C/C++/Java/Python данная операция обозначается как « | », а в Pascal как «or».

Помогите Рику и Морти посчитать, сколькими способами они могут выбрать две различные горы так, что если они начнут орать с этих гор, ор их будет слышен выше этих гор и всех гор между ними. Формально говоря, требуется вычислить, сколько существует таких пар l и r $(1 \le l < r \le n)$, что побитовое ИЛИ всех высот гор на отрезке от l до r включительно строго больше высоты любой горы на этом отрезке.

Входные данные
В первой строке содержится целое число n $(1 \le n \le 200\,000)$ — количество гор в хребте. В следующей строке содержатся n целых чисел a_i $(0 \le a_i \le 10^9)$ — высоты гор в порядке, в котором они следуют в хребте.

Выходные данные
Выведите одно число — искомое количество способов выбрать две различные горы.

входные данные
5 3 2 1 6 5
выходные данные
8

входные данные
4 3 3 3 3
выходные данные
0

В первом примере все искомые способы — это пары гор со следующими номерами (горы нумеруются с единицы):

$$(1, 4), (1, 5), (2, 3), (2, 4), (2, 5), (3, 4), (3, 5), (4, 5)$$

Во втором примере искомых пар не существует, поскольку для любой пары гор высота ора с них равна 3, и эта высота равна высоте любой из гор, следовательно она не выше их.

4A. Set

2 секунды, 256 мегабайт

Реализуйте множество с использованием хеш-таблицы.

В данной задаче запрещено пользоваться стандартной библиотекой языка (классами `std::set`, `std::unordered_set` и подобными в C++ и их аналогами в других языках программирования).

Входные данные
Входные данные содержат описание операций, которые необходимо выполнить. Каждая строка содержит одну из следующих операций:

- `insert` x — добавить элемент x в множество. Если элемент уже есть в множестве, не нужно ничего делать.
- `delete` x — удалить элемент x из множества. Если элемента нет в множестве, не нужно ничего делать.
- `exists` x — если элемент x есть в множестве, необходимо вывести «`true`». В противном случае необходимо вывести «`false`».

Все числа, используемые в запросах целые и не превосходят 10^9 по модулю.

Выходные данные
Для каждой операции `exists` выведите результат ее работы.

входные данные
insert 2 insert 5 insert 3 exists 2 exists 4 insert 2 delete 2 exists 2
выходные данные
true false false

4B. LinkedHashMap

2 секунды, 256 мегабайт

Реализуйте LinkedHashMap с использованием хеш-таблицы.

В данной задаче запрещено пользоваться стандартной библиотекой языка (классами `std::map`, `std::unordered_map` и подобными в C++ и их аналогами в других языках программирования).

Входные данные
Входные данные содержат описание операций, которые необходимо выполнить. Каждая строка содержит одну из следующих операций:

- `put` x y — добавить в соответствие ключу x значение y. Если такой ключ уже есть, значение нужно изменить на y.
- `delete` x — удалить ключ x. Если элемента нет, не нужно ничего делать.
- `get` x — если ключ x есть, необходимо вывести соответствующее ему значение. В противном случае необходимо вывести «`none`».
- `prev` x — вывести значение, соответствующее ключу, который был вставлен позже всех, но до ключа x. Если такого ключа нет, или ключа x нет, необходимо вывести «`none`».
- `next` x — вывести значение, соответствующее ключу, который был вставлен раньше всех, но после ключа x. Если такого ключа нет, или ключа x нет, необходимо вывести «`none`».

Все ключи и значения — строки из латинских букв длиной не более 20 символов.

Выходные данные
Для каждой операции `get`, `prev` и `next` выведите результат ее работы.

входные данные
put zero a put one b put two c put three d put four e get two prev two next two delete one delete three get two prev two next two next four
выходные данные
c b d c a e none

4C. Идеальное хеширование

2 секунды, 256 мегабайт

Вам задана архитектура компьютера, в рамках которой от вас требуется сделать программу, которая является идеальным хешом для заданного вам множества целых чисел.

Опишем архитектуру компьютера. У него есть 4 мебибайта = 4×2^{20} памяти, доступной только для чтения, содержащей программу, которую будет исполнять компьютер, и данные. Процессор компьютера содержит 256 регистров с названиями от `r0` до `r255`, каждый из них может сохранить 32-битное целое число. Перед исполнением программы значение каждого регистра кроме `r0` равно 0, а в `r0` содержится входные данные программы.

У компьютера есть специальный регистр, который называется instruction pointer. Программа хранится в памяти и изначально значение регистра `instruction pointer` равно 0. Выполнение программы проходит по следующему сценарию. Сначала читается номер инструкции, который хранится в памяти по адресу значения `instruction pointer`, после чего читаются аргументы инструкции и инструкция выполняется. Если во время выполнения инструкции не случился `jump`, то значение `instruction pointer` увеличивается на размер прочитанной инструкции, включая аргументы.

Программа должна быть отображением из множества заданных n целых чисел $A = \{a_1, a_2, \ldots, a_n\}$ в множество целых чисел от 0 до $2n - 1$. Она должна завершаться после не более чем 30 выполненных инструкций. Отображение не должно иметь коллизий. Формально, надо сделать программу в этой архитектуре, которая удовлетворяет следующим критериям:

- Если программе на вход подать одно из чисел из множества A, то она должна завершиться за не более чем 30 инструкций и вывести целое число от 0 до $2n - 1$.
- Для любых двух различных чисел a_i и a_j из множества A значения на выходе должны быть различны.

Вам задана некоторая документация.

- Все регистры содержат 32-битные целые числа
- Для сложения, вычитания и битовых операций не важно, числа знаковые или нет
- Команды `jump` с условием представляют числа как знаковые
- Результатом умножения или деления является знакового 64-битное число, которое сохраняется в два регистра. Регистр, который получает младшую часть результата сохраняет его как беззнаковое число
- Деление и взятие остатка принимают делимое в качестве 64-битных чисел, младшая часть интерпретируется как беззнаковое число, а старшая — как знаковое. Делитель — знаковое число.
- Деление и взятие остатка знаковых чисел производится так же, как это делается в архитектуре x86.
- Все числа в памяти сохраняются в порядке от младших байт к старшим (little endian).

От вас требуется понять, что должна содержать память, чтобы исполнялась программа, удовлетворяющая условиям, описанным выше.

Для любых входных данных из множества A ваша программа не должна пытаться читать данные извне 4МБ-диапазона и не должна делить на ноль. Значение регистра `instruction pointer` так же должно быть внутри 4МБ-диапазона.

Инстр.	Размер	Opcode	Что делает инструкция
nop	1 байт	`00`	Ничего
add	4 байта	`01 r1 r2 r3`	`r3 := r1 + r2`
sub	4 байта	`02 r1 r2 r3`	`r3 := r1 - r2`
mul	5 байт	`03 r1 r2 r3 r4`	`(r3, r4) := r1 * r2` r3 получает младшие биты, а r4 — старшие биты произведения
div	6 байт	`04 r1 r2 r3 r4 r5`	`(r3, r4) := (r1, r5) div r2` r1 содержит младшие биты, r5 содержит старшие биты делимого, r3 получает младшие биты, а r4 — старшие биты частного
mod	5 байт	`05 r1 r2 r3 r4`	`r3 := (r1, r4) mod r2`, r1 содержит младшие биты, r4 содержит старшие биты делимого
and	4 байта	`10 r1 r2 r3`	`r3 := r1 and r2`
or	4 байта	`11 r1 r2 r3`	`r3 := r1 or r2`
xor	4 байта	`12 r1 r2 r3`	`r3 := r1 xor r2`
neg	2 байта	`20 r1`	`r1 := -r1`
not	2 байта	`21 r1`	`r1 := ~r1`
load	3 байта	`30 r1 r2`	`r1 := memory[r2]` 4 байта, начиная с адреса r2 копируются в регистр r1, первым копируется младший байт
put	6 байт	`31 r1 b0 b1 b2 b3`	`r1 := (b0, b1, b2, b3)` здесь b0 — младший байт числа, положенного в регистр, а b3 — старший
jmp	5 байт	`40 b0 b1 b2 b3`	Сделать `jump` к инструкции (b0, b1, b2, b3)
jz	6 байт	`41 r1 b0 b1 b2 b3`	Если `r1 == 0` сделать `jump` к инструкции (b0, b1, b2, b3)
jnz	6 байт	`42 r1 b0 b1 b2 b3`	Если `r1 != 0` сделать `jump` к инструкции (b0, b1, b2, b3)
jg	6 байт	`43 r1 b0 b1 b2 b3`	Если `r1 > 0` сделать `jump` к инструкции (b0, b1, b2, b3)
jge	6 байт	`44 r1 b0 b1 b2 b3`	Если `r1 >= 0` сделать `jump` к инструкции (b0, b1, b2, b3)
jl	6 байт	`45 r1 b0 b1 b2 b3`	Если `r1 < 0` сделать `jump` к инструкции (b0, b1, b2, b3)
jle	6 байт	`46 r1 b0 b1 b2 b3`	Если `r1 <= 0`сделать `jump` к инструкции (b0, b1, b2, b3)
ret	1 байт	`ff`	Завершить программу. Выходные данные содержатся в регистре `r0`

Входные данные
Первая строка содержит целое число n ($1 \le n \le 100\,000$).

Вторая строка содержит n различных целых чисел $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$).

Выходные данные
Выведите данные, содержащиеся в памяти компьютера. Вам разрешается вывести любое число байт от 1 до $4\,194\,304$. Остальная часть памяти заполнится нулями. Каждый байт должен быть выведен как шестнадцатеричное число из двух цифр. Не печатайте пробелы.

Пожалуйста, выводите только требующийся префикс памяти вашей программы: не выводите конечные нули, чтобы тестирование проходило быстрее.

входные данные
3 2 3 9
выходные данные
3101040000000500010003ff

входные данные
2 6 10
выходные данные
300000ff0000000000001000000

4D. Хеш-код

2 секунды, 256 мегабайт

Согласно документации стандартной библиотеки Java, хеш-код для строки вычисляется как:

$$s[0]\cdot 31^{n-1} + s[1]\cdot 31^{n-2} + \ldots + s[n-1]$$

Где $s[i]$ — это i-й символ строки, n длина строки. Для вычисления используются целые 32-битные числа в форме дополнения до двух.

Вы собираетесь взломать сервера одной известной компании. Чтобы вы смогли выполнить атаку, вам нужны k различных строк, которые имеют одинаковые хеш-коды. К сожалению, сервера этой компании не принимают строки запроса, содержащие буквы отличные от английских букв нижнего и верхнего регистров.

Напишите программу, которая генерирует такие строки.

Входные данные
Первая строка содержит целое число k — количество необходимых строк запроса для генерации ($2 \le k \le 1000$).

Выходные данные
Необходимо вывести k различных непустых строк, каждая из которых имеет длину не более 1000 символов. Все строки должны состоять только из английских букв верхнего или нижнего регистров и иметь равный хеш-код.

входные данные
4
выходные данные
edHs mENAGeS fEHs edIT

4E. С днём рождения!

2 секунды, 256 мегабайт

В этой задаче требуется найти коллизию при полиномиальном хешировании строк, состоящих из маленьких букв английского алфавита.

Полиномиальный хеш строки имеет два параметра: множитель p и модуль q. Для пустой строки ε значение хеш-функции $h(\varepsilon) = 0$, а для любой строки S и любого символа c хеш-функция рекуррентно определяется как $h(S + c) = (h(S) \cdot p + \text{code}(c)) \bmod q$. Здесь $\text{code}(c)$ — это ASCII-код символа c. Как известно, коды маленьких букв английского алфавита идут подряд: $\text{code}(\text{'a'}) = 97$, $\text{code}(\text{'b'}) = 98, \ldots, \text{code}(\text{'z'}) = 122$. Можно выписать и нерекуррентную формулу: если строка $S = s_1 s_2 \ldots s_n$, то $h(S) = (\text{code}(s_1) \cdot p^{n-1} + \text{code}(s_2) \cdot p^{n-2} + \ldots + \text{code}(s_n) \cdot p^0) \bmod q$.

По заданным числам p и q найдите две различные непустые строки A и B такие, что $h(A) = h(B)$.

Входные данные
Первая строка ввода содержит два целых числа p и q, разделённых пробелом — параметры функции хеширования ($0 < p < q \leqslant 2 \cdot 10^9 + 9$).

Выходные данные
В первых двух строках выведите две различные непустые строки A и B, для которых $h(A) = h(B)$. Строки должны состоять исключительно из маленьких букв английского алфавита (ASCII-коды 97–122) и иметь длину от 1 до $100\,000$ символов. Заметим, что длины строк не обязательно должны совпадать. Если возможных ответов несколько, разрешается вывести любой из них.

входные данные
31 47
выходные данные
aa bq

входные данные
2 1000000007
выходные данные
rp nx

входные данные
179 1000000009
выходные данные
weeoutf hronndauw

5A. И снова сумма...

3 секунды, 256 мегабайт

Реализуйте структуру данных, которая поддерживает множество S целых чисел, с котором разрешается производить следующие операции:

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

Входные данные
Исходно множество S пусто. Первая строка входного файла содержит n — количество операций ($1 \leq n \leq 300\,000$). Следующие n строк содержат операции. Каждая операция имеет вид либо «`+ i`», либо «`? l r`». Операция «`? l r`» задает запрос $sum(l, r)$.

Если операция «`+ i`» идет во входном файле в начале или после другой операции «+», то она задает операцию $add(i)$. Если же она идет после запроса «?», и результат этого запроса был y, то выполняется операция $add((i + y) \bmod 10^9)$.

Во всех запросах и операциях добавления параметры лежат в интервале от 0 до 10^9.

Выходные данные
Для каждого запроса выведите одно число — ответ на запрос.

входные данные
6 + 1 + 3 + 3 ? 2 4 + 1 ? 2 4
выходные данные
3 7

5B. Вперёд!

3 секунды, 256 мегабайт

{256 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 которых звучит так: «Рядовые с l_i по l_j — вперёд!»

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

Например, если в какой-то момент солдаты стоят в порядке $2, 3, 6, 1, 5, 4$, то после приказа «Рядовые с 2 по 4 — 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.

Входные данные
В первой строке входного файла указаны числа n и m ($2 \leqslant n \leqslant 100\,000$, $1 \leqslant m \leqslant 100\,000$) — число солдат и число приказов. Следующие m строк содержат приказы в виде двух целых чисел: l_i и r_i ($1 \leqslant l_i \leqslant r_i \leqslant n$).

Выходные данные
Выведите в выходной файл n целых чисел — порядок, в котором будут стоять солдаты после исполнения всех приказов.

входные данные
6 3 2 4 3 5 2 2
выходные данные
1 4 5 2 3 6

5C. Переворот

2 секунды, 64 мегабайта

Дан массив. Надо научиться обрабатывать два типа запросов.

- `1 L R` - перевернуть отрезок [L, R]
- `2 L R` - найти минимум на отрезке [L, R]

Входные данные
Первая строка файла содержит два числа n, m. ($1 \leqslant n, m \leqslant 10^5$) Во второй строке находится n чисел a_i ($1 \leqslant a_i \leqslant 10^9$)- исходный массив. Остальные m строк содержат запросы, в формате описанном в условии. Для чисел L,R выполняется ограничение ($1 \leqslant L \leqslant R \leqslant n$).

Выходные данные
На каждый запрос типа 2, во входной файл выведите ответ на него, в отдельной строке.

входные данные
10 7 5 3 2 3 12 6 7 5 10 12 2 4 9 1 4 6 2 1 8 1 1 8 1 8 9 2 1 7 2 3 6

выходные данные
3 2 2 2

5D. Своппер

1 секунда, 256 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 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 обсчёт может затянуться надолго, корпорация «МИФ» попросила Вас решить проблему со своппером и промоделировать ЭТО эффективно.

Входные данные

Во входном файле заданы один или несколько тестов. В первой строке каждого теста записаны число N — длина последовательности и число M — число операций ($1 \le N,\ M \le 100\,000$). Во второй строке теста 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, и `2` a_i b_i — запрос второго типа. Сумма всех N и M по всему файлу не превосходит $200\,000$. Файл завершается строкой из двух нулей. Гарантируется, что $x_i < y_i$, а $a_i \le b_i$.

Выходные данные

Для каждого теста выведите ответы на запросы второго типа, как показано в примере. Разделяйте ответы на тесты пустой строкой.

входные данные
5 5 1 2 3 4 5 1 2 5 2 2 4 1 1 4 2 1 3 2 4 4 0 0

выходные данные
Swapper 1: 10 9 2

5E. Викторина «Три топора»

2 секунды, 1024 мегабайта

В викторине «Три топора» участвуют n человек, у каждого из которых есть два типа ресурсов — *очки* и *монеты*. Изначально у каждого участника 0 очков и 0 монет.

Викторина состоит из q вопросов. Участник, который первым правильно ответил на i-й вопрос, получает p_i очков. Интересно то, что по правилам викторины, величина p_i может быть отрицательной.

После каждого вопроса строится рейтинговая таблица участников, в которой участники упорядочиваются по невозрастанию количества очков. После этого **каждый** участник получает некоторое количество монет по следующему принципу. Если до i-го вопроса некоторый участник находится на *месте* с номером a, а после i-го вопроса он находится на месте b, данный участник получает $|a-b|$ монет. Место участника вычисляется как 1 плюс количество участников, которые имеют строго больше очков, чем данный участник. Например, в начале викторины все участники имеют 0 очков, поэтому все они занимают 1-е место.

Вам в руки попала запись, которая велась по ходу викторины. Для каждого вопроса запись содержит номер участника a_i, который первым ответил на i-й вопрос, а также количество очков p_i. От вас требуется выяснить, сколько монет заработал каждый участник по результатам всей викторины.

Входные данные

Первая строка содержит два целых числа n и q ($1 \le n, q \le 100\,000$) — количество участников викторины, а также количество вопросов.

Каждая из следующих q строк содержит два целых числа a_i и p_i ($1 \le a_i \le n$, $-10^9 \le p_i \le 10^9$) — номер участника, который первым ответил на i-й вопрос, а также прибавка к его количеству очков.

Выходные данные

Выведите n целых чисел, каждое в отдельной строке: в i-й строке выведите одно целое число — количество монет, заработанных участником с номером i по результатам всей викторины.

входные данные
3 7 2 -1 1 4 2 5 3 6 1 -7 3 -6 2 9

выходные данные
2 6 5

входные данные
9 5 2 10 2 -20 2 20 2 -20 2 20

выходные данные
5 32 5 5 5 5 5 5 5

входные данные
5 10 1 0 3 0 2 0 5 0 4 0 1 0 3 0 2 0 5 0 4 0

выходные данные
0 0 0 0 0

5F. Асхат и дерево

2 секунды, 256 мегабайт

Это интерактивная задача.

Дано двоичное дерево поиска размера n, в каждой вершине есть значение от 1 до n. Петух по имени Асхат каждый день нумерует вершины числами от 1 до n, даёт вам номер корня r и просит вас найти номер вершины со значением x.

Вы можете совершать запросы — по номеру вершины узнать значение в ней и номера её детей. Пусть максимальное расстояние от корня до вершины в i-й день равно d. Тогда в i-й день вы можете совершить не более, чем d запросов. За все дни вы можете совершить не более, чем 70000 запросов.

Также вы можете менять детей у каждой вершины. Пусть в i-й день длина пути от корня до вершины с номером x равна s. Тогда вам в i-й день разрешено сделать не более, чем s запросов вида «установить у вершины новых детей». За все дни вы можете совершить не более, чем 70000 запросов этого типа.

Протокол взаимодействия

В первой строке ввода даны числа n и q ($1 \le n \le 2000$, $1 \le q \le 2000$) — размер дерева и число запросов, соответственно.

Для каждого запроса даны числа r и x ($1 \le r, x \le n$) — номер корня дерева и значение, вершину с которым требуется найти. Вы не сможете считать эти числа для следующего запроса, пока не дадите ответ на текущий.

Чтобы обратиться к вершине с номером i выведите «`val i`» в отдельной строке. В ответ даются три числа val, L и R ($1 \le val \le n$, $0 \le L, R \le n$) — значение в этой вершине и номера левого и правого ребёнка, соответственно. В случае, если у вершины нет левого или правого ребёнка, $L = 0$ или $R = 0$, соответственно.

Чтобы поменять детей вершины с номером i на вершины с номером L и R выведите «`change i L R`» в отдельной строке. Чтобы у вершины с номером i не было левого или правого ребёнка, выведите 0 вместо L или R, соответственно. После выполнения этого запроса граф может перестать быть двоичным деревом поиска.

Если искомое значение находится в вершине с номером i и вы совершили все нужные изменения, выведите «`confirm i`» в отдельной строке. После этого вершины перенумеруются, а на ввод будет дан новый запрос. Если на момент выполнения этого запроса граф не является двоичным деревом поиска, вы получите вердикт «Wrong answer».

6A. LCA Problem

5 секунд, 256 мегабайт

Задано подвешенное дерево, содержащее n ($1 \le n \le 10^5$) вершин, пронумерованных от 0 до $n-1$. Требуется ответить на m ($1 \le m \le 10^6$) запросов о наименьшем общем предке для пары вершин.

Запросы генерируются следующим образом. Заданы числа a_1, a_2 и числа x, y, z. Числа $a_3, \ldots, a_{2m}$ генерируются следующим образом: $a_i = (x \cdot a_{i-2} + y \cdot a_{i-1} + z) \mod n$. Первый запрос имеет вид (a_1, a_2). Если ответ на $i-1$-й запрос равен v, то i-й запрос имеет вид $((a_{2i-1} + v) \mod n, a_{2i})$.

Входные данные

Первая строка содержит два числа: n и m. Корень дерева имеет номер 0.

Вторая строка содержит $n-1$ целых чисел, i-е из этих чисел равно номеру родителя вершины i. Третья строка содержит два целых числа в диапазоне от 0 до $n-1$: a_1 и a_2.

Четвертая строка содержит три целых числа: x, y, z, эти числа неотрицательны и не превосходят 10^9.

Выходные данные

Выведите в выходной файл сумму номеров вершин — ответов на все запросы.

входные данные
3 2 0 1 2 1 1 1 0
выходные данные
2

входные данные
1 2 0 0 1 1 1
выходные данные
0

6B. Дуумвират

1 секунда, 256 мегабайт

Вам дано дерево. В вершинах записаны числа. Нужно научиться находить сумму чисел на пути из v в u.

Входные данные

В первой строке записано число n — количество вершин дерева ($1 \le n \le 10^5$). Во сторой сроке записаны через пробел n чисел v_i ($|v_i| < 10^9$), задающие значения в вершинах. В следующих $n-1$ строках описаны ребра дерева. В $(i+2)$-й строке записаны номера вершин a_i, b_i ($1 \le a_i, b_i \le n$), означающие, что в дереве есть ребро из вершины a_i в вершину b_i.

Далее на отдельной строке записано число m — количество запросов ($1 \le m \le 10^5$). После этого идут m строк с описанием запросов, в $(n+2+i)$-й строке записаны через пробел числа x_i и y_i ($1 \le x_i, y_i \le n$).

Выходные данные

Для каждого запроса на отдельной строке требуется вывести сумму всех значений v_i по всем вершинам на пути из x_i в y_i.

входные данные
4 -9 -6 -1 9 1 2 3 1 4 1 6 1 2 3 2 2 3 4 2 4 3 2 1
выходные данные
-15 -16 -16 -6 -1 -15

6C. Самое дешевое ребро

2 секунды, 256 мегабайт

Дано подвешенное дерево с корнем в первой вершине. Все ребра имеют веса (стоимости). Вам нужно ответить на M запросов вида "найти у двух вершин минимум среди стоимостей ребер пути между ними".

Входные данные

В первой строке файла записано одно числ — n (количество вершин).

В следующих $n-1$ строках записаны два числа — x и y. Число x на строке i означает, что x — предок вершины i, y означает стоимость ребра.

$x < i,\ |y| \le 10^6$.

Далее m запросов вида (x, y) — найти минимум на пути из x в y ($x \neq y$).

Ограничения: $2 \leq n \leq 5 \cdot 10^4$, $0 \leq m \leq 5 \cdot 10^4$.

Выходные данные

Выведите m ответов на запросы.

входные данные
5 1 2 1 3 2 5 3 2 2 2 3 4 5
выходные данные
2 2

6D. Чип и Дейл в лабиринте

1 секунда, 256 мегабайт

Чип и Дейл спешат на помощь! Но внимательные зрители знают, что помощь как правило нужна самим Чипу и Дейлу, поэтому сегодня вам надо будет сыграть роль сообразительной Гаечки. Итак, Чип и Дейл снова попали в лапы к Толстопузу. Кот очень не любит грызунов и поэтому приготовил им изощренное испытание. Он собирается поместить их в лабиринт и посмотреть смогут ли они из него выбраться. Лабиринт представляет собой дерево, в котором каждое ребро имеет одно направление. Гаечка подслушала разговор Толстопузу со своими сообщниками и теперь знает несколько возможных вариантов: в какую точку лабиринта поместят её друзей, и где будет выход. Для каждого такого варианта она хочет понять, смогут ли Чип и Дейл найти выход, или нет.

Входные данные

В первой строке входного файла записано число n $(1 \leq n \leq 10^5)$ — число вершин в дереве. В следующих $n - 1$ строках описаны ребра дерева. В $i + 1$ строке файла записаны два числа $a_i, b_i (1 \leq a_i, b_i \leq n)$, означающие, что существует ребро из a_i в b_i.

Далее записано число m $(1 \leq m \leq 10^5)$ — число запросов. После этого идет описание запросов, каждый запрос в новой строке. Для каждого запроса задается $x_i, y_i (1 \leq x_i, y_i \leq n)$ — точка, в которую поместят Чипа и Дейла, и выход из лабиринта соответственно.

Выходные данные

Для каждого запроса надо в отдельной строке вывести Yes, если бурундуки смогут найти выход, и No иначе.

входные данные
4 1 2 3 1 4 1 6 1 2 3 2 2 3 4 2 4 3 2 1
выходные данные
Yes Yes No Yes No No

6E. А и В и аудитории

2 секунды, 256 мегабайт

А и В готовятся к олимпиадам про программированию.

Университет, в котором учатся А и В, представляет собой множество аудиторий, соединенных между собой переходами. Всего в университете n аудиторий, соединённых n - 1 переходом таким образом, что из любой аудитории можно дойти до любой по переходам. Аудитории пронумерованы от 1 до n.

Каждый день А и В пишут контесты в некоторых аудиториях своего университета, и после каждого контеста они собираются вместе в одной аудитории и обсуждают задачи. А и В хотят, чтобы расстояния от аудитории, где они будут обсуждать задачи, до аудиторий, где они пишут контесты, были равны. Расстоянием между двумя аудиториями является количество ребер на кратчайшем пути между ними.

Так как каждый день они пишут контесты в новых аудиториях, то они попросили вас помочь им найти количество возможных аудиторий для обсуждения задач на каждый из ближайших m дней.

Входные данные

В первой строке задано целое число n $(1 \leq n \leq 10^5)$ — количество аудиторий в университете.

В следующих n - 1 строках описаны переходы. В i-й из этих строк $(1 \leq i \leq n - 1)$ записаны два целых числа a_i и b_i $(1 \leq a_i, b_i \leq n)$, обозначающих, что i-й переход соединяет аудитории с номерами a_i и b_i.

В следующей строке записано целое число m $(1 \leq m \leq 10^5)$ — количество запросов.

Следующие m строк описывают сами запросы. В j-й из этих строк $(1 \leq j \leq m)$ записаны два целых числа x_j и y_j $(1 \leq x_j, y_j \leq n)$, обозначающих, что в j-й день А будет писать контест в аудитории с номером x_j, В — в аудитории с номером y_j.

Выходные данные

В i-й $(1 \leq i \leq m)$ строке выведите количество аудиторий, равноудаленных от аудиторий, в которых А и В пишут контест в i-й день.

входные данные
4 1 2 1 3 2 4 1 2 3
выходные данные
1

входные данные
4 1 2 2 3 2 4 2 1 2 1 3
выходные данные
0 2

В первом примере из условия на одинаковом расстоянии от аудиторий с номерами 2 и 3 находится только одна аудитория — аудитория с номером 1.

6F. Камил и проведение стрима

4 секунды, 768 мегабайт

Камил любит стримить видео по спортивному программированию. Его канал на MeTube недавно набрал 100 миллионов подписчиков. Чтобы отпраздновать это, он выложил видео с интересной задачей, которую он пока не может решить. Можете ли вы помочь ему?

Вам дано дерево — связный неориентированный граф состоящий из n вершин и $n - 1$ ребер. Дерево подвешено за вершину 1. Вершина u является *предком* v если она лежит на кратчайшем пути между корнем и v. В частности, вершина является предком себя.

У каждой вершины v есть *красота* x_v — неотрицательное целое число не большее 10^{12}. Это позволяет нам определить красоту пути. Пусть u предок v. Тогда определим красоту $f(u, v)$ как наибольший общий делитель красот всех вершин на кратчайшем пути между u и v. Формально, если $u = t_1, t_2, t_3, \ldots, t_k = v$ — вершины на кратчайшем пути между u и v, тогда $f(u, v) = \gcd(x_{t_1}, x_{t_2}, \ldots, x_{t_k})$. Тут $\gcd$ обозначает наибольший общий делитель множества чисел. В частности, $f(u, u) = \gcd(x_u) = x_u$.

Ваша задача найти сумму

$$\sum_{u \text{ предок } v} f(u, v).$$

Так как ответ может быть слишком большим, выведите его по модулю $10^9 + 7$.

Обратите внимание что для любого y, $\gcd(0, y) = \gcd(y, 0) = y$. В частности, $\gcd(0, 0) = 0$.

Входные данные

В первой строке записано одно целое число n ($2 \le n \le 100\,000$) — количество вершин в дереве.

В следующей строке записаны n целых чисел $x_1, x_2, \ldots, x_n$ ($0 \le x_i \le 10^{12}$). Значение x_v обозначает красоту вершины v.

В следующих $n - 1$ строках описаны ребра дерева. Каждая из них содержит два целых числа a, b ($1 \le a, b \le n$, $a \ne b$) — вершины соединенные ребром.

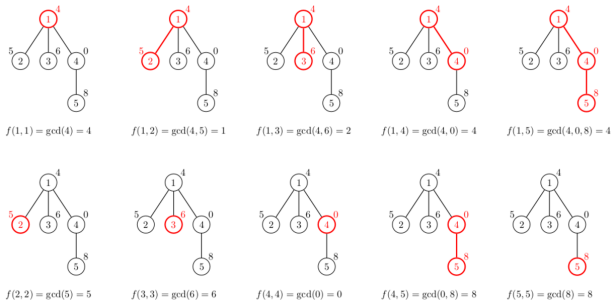
Выходные данные

Выведите сумму красот всех таких путей (u, v), что u предок v. Эта сумма должна быть выведена по модулю $10^9 + 7$.

входные данные
5 4 5 6 0 8 1 2 1 3 1 4 4 5
выходные данные
42

входные данные
7 0 2 3 0 0 0 0 1 2 1 3 2 4 2 5 3 6 3 7
выходные данные
30

На следующей иллюстрации изображены все 10 возможных путей, в которых один из концов является предком другого. Сумма красот всех этих путей равна 42:



7A. Пещеры и туннели

2 секунды, 256 мегабайт

После посадки на Марс учёные нашли странную систему пещер, соединённых туннелями. И учёные начали исследовать эту систему, используя управляемых роботов. Было обнаружено, что существует ровно один путь между каждой парой пещер. Но потом учёные обнаружили специфическую проблему. Иногда в пещерах происходят небольшие взрывы. Они вызывают выброс радиоактивных изотопов и увеличивают уровень радиации в пещере. К сожалению, роботы плохо выдерживают радиацию. Но для исследования они должны переместиться из одной пещеры в другую. Учёные поместили в каждую пещеру сенсор для мониторинга уровня радиации. Теперь они каждый раз при движении робота хотят знать максимальный уровень радиации, с которым придётся столкнуться роботу во время его перемещения. Как вы уже догадались, программу, которая это делает, будете писать вы.

Входные данные

Первая строка входного файла содержит одно целое число n ($1 \le n \le 100\,000$) — количество пещер.

Следующие $n - 1$ строк описывают туннели. Каждая из этих строк содержит два целых числа — a_i и b_i ($1 \le a_i, b_i \le N$), описывающие туннель из пещеры с номером a_i в пещеру с номером b_i.

Следующая строка содержит целое число q ($1 \le q \le 100\,000$), означающее количество запросов.

Далее идут q запросов, по одному на строку. Каждый запрос имеет вид «c u v», где c — символ «`I`» либо «`G`», означающие тип запроса (кавычки только для ясности).

В случае запроса «`I`» уровень радиации в u-й пещере ($1 \le u \le n$) увеличивается на v ($0 \le v \le 10\,000$). В случае запроса «`G`» ваша программа должна вывести максимальный уровень радиации на пути между пещерами с номерами u и v ($1 \le u, v \le N$) после всех увеличений радиации (запросов «`I`»), указанных ранее.

Предполагается, что изначальный уровень радиации равен 0 во всех пещерах, и он никогда не уменьшается со временем (потому что период полураспада изотопов много больше времени наблюдения).

Выходные данные

Для каждого запроса «`G`» выведите одну строку, содержащую максимальный уровень радиации.

входные данные
4 1 2 2 3 2 4 6 I 1 1 G 1 1 G 3 4 I 2 3 G 1 1 G 3 4
выходные данные
1 0 1 3

7B. Прибавление на пути

4 секунды, 256 мегабайт

Задано дерево. В каждой вершине есть значение, изначально все значения равны нулю. Требуется обработать запрос прибавления на пути и запрос значения в вершине.

Входные данные

В первой строке задано целое число n — число вершин в дереве ($1 \le n \le 3 \cdot 10^5$).

В следующих $n - 1$ строках заданы ребра дерева: по два целых числа v и u в строке — номера вершин, соединенных ребром ($1 \le v, u \le n$).

В следующей строке задано целое число m — число запросов ($1 \le m \le 5 \cdot 10^5$).

Следующие m строк содержат запросы в одном из двух форматов:

- `+ v u d` — прибавить число d во все значения в вершинах на пути от v до u ($1 \le v, u \le n$; $1 \le d \le 10^9$);
- `? v` — вывести значение в вершине v ($1 \le v \le n$).

Выходные данные

Выведите ответы на все запросы.

входные данные
5 1 2 1 3 3 4 3 5 5 + 2 5 1 ? 3 + 1 1 2 ? 1 ? 3

выходные данные
1 3 1