15 - Персистентные структуры данных

A. Персистентный стек

2 секунды, 256 мегабайт

Реализуйте персистентный стек.

Входные данные

Первая строка содержит количество действий n ($1 \le n \le 200\,000$). В строке номер $i + 1$ содержится описание действия i:.

- `t m` — добавить в конец стека номер t ($0 \le t < i$) число m ($0 < m \le 1000$);
- `t 0` — удалить последний элемент стека номер t ($0 \le t < i$). Гарантируется, что стек t не пустой.

В результате действия i, описанного в строке $i + 1$, создается стек номер i. Изначально имеется пустой стек с номером ноль.
Все числа во входном файле целые.

Выходные данные

Для каждой операции удаления выведите удаленный элемент на отдельной строке.

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

B. Персистентный массив

2 секунды, 256 мегабайт

Дан массив (вернее, первая, начальная его версия).

Нужно уметь отвечать на два запроса:

- $a_i[j] = x$ — создать из i-й версии новую, в которой j-й элемент равен x, а остальные элементы такие же, как в i-й версии.
- `get` $a_i[j]$ — сказать, чему равен j-й элемент в i-й версии.

Входные данные

Количество чисел в массиве N ($1 \le N \le 10^5$) и N элементов массива. Далее количество запросов M ($1 \le M \le 10^5$) и M запросов. Формат описания запросов можно посмотреть в примере. Если уже существует K версий, новая версия получает номер $K + 1$. И исходные, и новые элементы массива — целые числа от 0 до 10^9. Элементы в массиве нумеруются числами от 1 до N.

Выходные данные

На каждый запрос типа `get` вывести соответствующий элемент нужного массива.

входные данные
6 1 2 3 4 5 6 11 create 1 6 10 create 2 5 8 create 1 5 30 get 1 6 get 1 5 get 2 6 get 2 5 get 3 6 get 3 5 get 4 6 get 4 5
выходные данные
6 5 10 5 10 8 6 30

K-я порядковая статистика на отрезке

4.0 с, 1024 МБ

Дан массив из N неотрицательных чисел, строго меньших 10^9. Вам необходимо ответить на несколько запросов о величине k-й порядковой статистики на отрезке $[l, r]$.

Входные данные

Первая строка содержит число N ($1 \le N \le 450\,000$) — размер массива.

Вторая строка может быть использована для генерации a_i — начальных значений элементов массива. Она содержит три числа a_1, l и m ($0 \le a_1, l, m < 10^9$); для i от 2 до N

$$a_i = (a_{i-1} \cdot l + m) \bmod 10^9.$$

В частности, $0 \le a_i < 10^9$.

Третья строка содержит одно целое число B ($1 \le B \le 1000$) — количество групп запросов.

Следующие B строк описывают одну группу запросов. Каждая группа запросов описывается 10 числами. Первое число G обозначает количество запросов в группе. Далее следуют числа x_1, l_x и m_x, затем y_1, l_y и m_y, затем, k_1, l_k и m_k ($1 \le x_1 \le y_1 \le N$, $1 \le k_1 \le y_1 - x_1 + 1$, $0 \le l_x, m_x, l_y, m_y, l_k, m_k < 10^9$). Эти числа используются для генерации вспомогательных последовательностей x_g и y_g, а также параметров запросов i_g, j_g и k_g ($1 \le g \le G$)

$$\begin{aligned}
x_g &= ((i_{g-1} - 1) \cdot l_x + m_x) \bmod N) + 1, && 2 \le g \le G \\
y_g &= ((j_{g-1} - 1) \cdot l_y + m_y) \bmod N) + 1, && 2 \le g \le G \\
i_g &= \min(x_g, y_g), && 1 \le g \le G \\
j_g &= \max(x_g, y_g), && 1 \le g \le G \\
k_g &= (((k_{g-1} - 1) \cdot l_k + m_k) \bmod (j_g - i_g + 1)) + 1, && 2 \le g \le G
\end{aligned}$$

Сгенерированные последовательности описывают запросы, g-й запрос состоит в поиске k_g-го по величине числа среди элементов отрезка $[i_g, j_g]$.

Суммарное количество запросов не превосходит $600\,000$.

Выходные данные

Выведите единственное число — сумму ответов на запросы.

с. k-я порядковая статистика на отрезке
5 1 1 1 5 1 1 0 0 3 0 0 2 0 0 1 2 0 0 5 0 0 3 0 0 1 1 0 0 5 0 0 5 0 0 1 3 0 0 3 0 0 1 0 0 1 1 0 0 4 0 0 1 0 0
выходные данные
15

D. Менеджер памяти

10 секунд, 1024 мегабайта

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:

- `copy(a, b, l)` — скопировать отрезок длины $[a, a + l - 1]$ в $[b, b + l - 1]$
- `sum(l, r)` — посчитать сумму элементов массива на отрезке $[l, r]$
- `print(l, r)` — напечатать элементы с l по r, включительно

Вы являетесь разработчиком своей операционной системы, и Вы, безусловно, не можете обойтись без инновационных технологий. Вам необходимо реализовать точно такой же менеджер памяти.

Входные данные

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

Во второй строке содержатся четыре числа $1 \le X_1, A, B, M \le 10^9 + 10$. С помощью них можно сгенерировать исходный массив чисел $X_1, X_2, \cdots, X_N$. $X_{i+1} = (A * X_i + B) \mod M$

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

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

- `cpy` a b l — для операции `copy`
- `sum` l r — для операции `sum` ($l \le r$)
- `out` l r — для операции `print` ($l \le r$)

Гарантируется, что суммарная длина запросов `print` не превышает $3\,000$. Также гарантируется, что все запросы корректны.

Выходные данные

Для каждого запроса *sum* или *print* выведите в выходной файл на отдельной строке результат запроса.

входные данные
6 1 4 5 7 7 out 1 6 cpy 1 3 2 out 1 6 sum 1 4 cpy 1 2 4 out 1 6 sum 1 6
выходные данные
1 2 6 1 2 6 1 2 1 2 2 6 6 1 1 2 1 2 6 13

E. Урны и шары

4 секунды, 512 МБ

Пусть у вас есть n урн, в каждой из которых лежит по одному шарику. Урна с номером i содержит шарик под номером i. У вас есть специальное устройство, которое позволяет перемещать шарики. Им чрезвычайно просто пользоваться: сначала вы выбираете некоторый отрезок последовательных урн. После этого вы выбираете некоторый другой отрезок последовательных урн такой же длины, как и исходный, и затем шарики из урн первого отрезка перемещаются в соответствующие урны второго отрезка.

Дана последовательность перемещений. Установите, в какой урне окажется каждый шарик.

Входные данные

Первая строка входных данных содержит два числа n и m — число урн и число перемещений, соответственно ($1 \le n \le 100\,000$, $1 \le m \le 50\,000$). Каждая из следующих m строк содержит три числа $count_i, from_i$ и to_i, которые означают одновременное перемещение всех шариков из урны $from_i$ в урну to_i, всех шариков из урны $from_i + 1$ в урну $to_i + 1$, …, всех шариков из урны $from_i + count_i - 1$ в урну $to_i + count_i - 1$ ($1 \le count_i, from_i, to_i \le n$, $\max(from_i, to_i) + count_i \le n + 1$).

Выходные данные

Выведите n чисел — итоговые позиции каждого шарика.

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