Задача 1. Сортировка деревом поиска

Источник:	базовая*
Имя входного файла:	`input.bin`
Имя выходного файла:	`output.bin`
Ограничение по времени:	2 секунды
Ограничение по памяти:	разумное

В первых четырёх байтах входного файла задано целое число N — количество чисел в массиве A. Далее идут N четырёхбайтовых целых чисел — содержимое массива A. Размер массива лежит в диапазоне: $0 \leqslant N \leqslant 500\,000$.

Требуется отсортировать массив A по неубыванию, используя **дерево поиска**. Учтите, что в исходном массиве может быть много одинаковых элементов. Кроме того, элементы массива могут быть изначально выстроены в каком-то фиксированном порядке.

В выходной файл нужно вывести ровно N четырёхбайтовых целых чисел: содержимое массива A после сортировки.

Пример

input.bin
0A 00 00 00 1F 00 00 00 F2 FF FF FF 06 00 00 00 04 00 00 00 26 00 00 00 FD FF FF FF 1E 00 00 00 F6 FF FF FF 0A 00 00 00 F4 FF FF FF
output.bin
F2 FF FF FF F4 FF FF FF F6 FF FF FF FD FF FF FF 04 00 00 00 06 00 00 00 0A 00 00 00 1E 00 00 00 1F 00 00 00 26 00 00 00

Задача 2. Динамический поиск

Источник:	основная*
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда*
Ограничение по памяти:	разумное

Имеется множество целых чисел X, изначально оно пустое. Нужно выполнить M заданных операций над этим множеством.

Есть три типа операций:

1. `add` v — добавить число v в множество X. Если такого числа ещё не было в множестве, надо его добавить и напечатать слово `added`. Если такое число уже есть в множестве, нужно напечатать слово `dupe` и ничего не делать.

2. `remove` v — удалить число v из множества X. Если такое число есть в множестве, нужно его удалить и напечатать слово `removed`. А если такого числа нет, нужно напечатать слово `miss` и ничего не делать.

3. `lower` v — найти минимальное число в множестве X, которое больше или равно заданному v (т.е. `lower_bound`). Если такое число в множестве есть, нужно напечатать в файл. А если его нет, то есть если v больше всех чисел множества X, то нужно напечатать `###` (три символа решётки, ASCII 35).

Внимание: операции нужно выполнять в режиме "online": считывать операцию из файла разрешается только после того, как все предыдущие операции уже выполнены.

Задачу **можно** решать **без дерева поиска** за время $O(M\sqrt{M}\log M)$.

Формат входных данных

В первой строке содержится целое число M — количество операций ($1 \leqslant M \leqslant 10^5$). В остальных M строках записаны операции в порядке их выполнения. Все числа v в файле целые и по абсолютной величине не превышают 10^9.

Формат выходных данных

Нужно вывести M строк, в каждой из которых требуется записать результат выполнения соответствующей операции.

Пример

input.txt	output.txt
16 add 7 add 3 add 5 add 5 add 10 add 7 remove 6 remove 5 add 5 remove 3 lower 2 lower 5 add 1 lower 0 lower 10 lower 15	added added added dupe added dupe miss removed added removed 5 5 added 1 10 ###

Задача 3. Динамический поиск+

Источник:	повышенной сложности*
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	1 секунда*
Ограничение по памяти:	разумное

Имеется множество целых чисел X, изначально оно пустое. Нужно выполнить M заданных операций над этим множеством.

Есть три типа операций:

1. `add` v — добавить число v в множество X. Если такого числа ещё не было в множестве, надо его добавить и напечатать слово `added`. Если такое число уже есть в множестве, нужно напечатать слово `dupe` и ничего не делать.

2. `remove` v — удалить число v из множества X. Если такое число есть в множестве, нужно его удалить и напечатать слово `removed`. А если такого числа нет, нужно напечатать слово `miss` и ничего не делать.

3. `lower` v — найти минимальное число в множестве X, которое больше или равно заданному v (т.е. `lower_bound`). Если такое число в множестве есть, нужно напечатать в файл. А если его нет, то есть если v больше всех чисел множества X, то нужно напечатать `###` (три символа решётки, ASCII 35).

Внимание: операции нужно выполнять в режиме "online": считывать операцию из файла разрешается только после того, как все предыдущие операции уже выполнены.

Задачу нужно решать **используя сбалансированное дерево поиска**.

Формат входных данных

В первой строке содержится целое число M — количество операций ($1 \leqslant M \leqslant 3 \cdot 10^5$). В остальных M строках записаны операции в порядке их выполнения. Все числа v в файле целые и по абсолютной величине не превышают 10^9.

Формат выходных данных

Нужно вывести M строк, в каждой из которых требуется записать результат выполнения соответствующей операции.

Пример

input.txt	output.txt
16 add 7 add 3 add 5 add 5 add 10 add 7 remove 6 remove 5 add 5 remove 3 lower 2 lower 5 add 1 lower 0 lower 10 lower 15	added added added dupe added dupe miss removed added removed 5 5 added 1 10 ###

Задача 4. sql join: дерево поиска

Источник:	повышенной сложности*
Имя входного файла:	`input.txt`
Имя выходного файла:	`output.txt`
Ограничение по времени:	2 секунды
Ограничение по памяти:	разумное

Данная задача является продолжением задачи «sql join». Прочитайте условие оригинальной задачи перед тем, как читать дальше!

В данной задаче необходимо использовать **дерево поиска** для ускорения соединения. Занесите все записи одной таблицы в бинарное дерево поиска (при этом нужно решить, что делать с равными ключами). Далее переберите все записи второй таблицы: для каждой из них можно найти в дереве поиска список всех записей с совпадающим именем.

Входные/выходные данные в этой задаче такие же, как в задаче «sql join». Ограничения такие же ($1 \leqslant N, M \leqslant 10^5$), за одним важным исключением:

В этой задаче **не** гарантируется, что $N \cdot M \leqslant 10^5$. Вместо этого гарантируется, что после соединения количество записей R не превышает 10^5.