y2018-2-2. Дерево поиска

A. Простое двоичное дерево поиска

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 512 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Реализуйте просто двоичное дерево поиска.

Входные данные

Входной файл содержит описание операций с деревом, их количество не превышает 100. В каждой строке находится одна из следующих операций:

- insert *X* — добавить в дерево ключ *X*. Если ключ *X* есть в дереве, то ничего делать не надо
- delete *X* — удалить из дерева ключ *X*. Если ключа *X* в дереве нет, то ничего делать не надо
- exists *X* — если ключ *X* есть в дереве выведите «true», если нет «false»
- next *X* — выведите минимальный элемент в дереве, строго больший *X*, или «none» если такого нет
- prev *X* — выведите максимальный элемент в дереве, строго меньший *X*, или «none» если такого нет

В дерево помещаются и извлекаются только целые числа, не превышающие по модулю 10^9.

Выходные данные

Выведите последовательно результат выполнения всех операций exists, next, prev. Следуйте формату выходного файла из примера.

Пример

входные данные
insert 2 insert 5 insert 3 exists 2 exists 4 next 4 prev 4 delete 5 next 4 prev 4
выходные данные
true false 5 3 none 3

B. Сбалансированное двоичное дерево поиска

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 512 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Реализуйте сбалансированное двоичное дерево поиска.

Входные данные

Входной файл содержит описание операций с деревом, их количество не превышает 10^5. В каждой строке находится одна из следующих операций:

- insert *X* — добавить в дерево ключ *X*. Если ключ *X* есть в дереве, то ничего делать не надо
- delete *X* — удалить из дерева ключ *X*. Если ключа *X* в дереве нет, то ничего делать не надо
- exists *X* — если ключ *X* есть в дереве выведите «true», если нет «false»
- next *X* — выведите минимальный элемент в дереве, строго больший *X*, или «none» если такого нет
- prev *X* — выведите максимальный элемент в дереве, строго меньший *X*, или «none» если такого нет

В дерево помещаются и извлекаются только целые числа, не превышающие по модулю 10^9.

Выходные данные

Выведите последовательно результат выполнения всех операций exists, next, prev. Следуйте формату выходного файла из примера.

Пример

входные данные
insert 2 insert 5 insert 3 exists 2 exists 4 next 4 prev 4 delete 5 next 4 prev 4
выходные данные
true false 5 3 none 3

C. Декартово дерево

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Вам даны пары чисел (a_i, b_i). Необходимо построить декартово дерево, такое что i-я вершина имеет ключи (a_i, b_i), вершины с ключом a_i образуют бинарное дерево поиска, а вершины с ключом b_i образуют кучу.

Входные данные

В первой строке записано число N — количество пар. Далее следует N ($1 \le N \le 300\,000$) пар (a_i, b_i). Для всех пар $|a_i|, |b_i| \le 1\,000\,000$. $a_i \neq a_j$ и $b_i \neq b_j$ для всех $i \neq j$.

Выходные данные

Если декартово дерево с таким набором ключей построить возможно, выведите в первой строке «YES», в противном случае выведите «NO». В случае ответа «YES» выведите N строк, каждая из которых должна описывать вершину. Описание вершины состоит из трёх чисел: номера предка, номера левого сына и номера правого сына. Если у вершины отсутствует предок или какой либо из сыновей, выведите на его месте число 0.

Если подходящих деревьев несколько, выведите любое.

Пример

входные данные
7 5 4 2 2 3 9 0 5 1 3 6 6 4 11
выходные данные
YES 2 3 6 0 5 1 1 0 7 5 0 0 2 4 0 1 0 0 3 0 0

Условие недоступно на русском языке

E. И снова сумма

ограничение по времени на тест: 3 секунды
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

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

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

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

Если операция «+ i» идет во входном файле в начале или после другой операции «+», то она задает операцию . Если же она идет после запроса «?», и результат этого запроса был y, то выполняется операция .

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.

Пример

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

K-й максимум

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 512 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Напишите программу, реализующую структуру данных, позволяющую добавлять и удалять элементы, а также находить k-й максимум.

Входные данные
Первая строка входного файла содержит натуральное число n — количество команд ($n \le 100\,000$). Последующие n строк содержат по одной команде каждая. Команда записывается в виде двух чисел c_i и k_i — тип и аргумент команды соответственно ($|k_i| \le 10^9$).
Поддерживаемые команды:

- +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.

Выходные данные
Для каждой команды нулевого типа в выходной файл должна быть выведена строка, содержащая единственное число — k_i-й максимум.

Пример

f. k-й максимум
11 +1 5 +1 3 +1 7 0 1 0 2 0 3 -1 5 +1 10 0 1 0 2 0 3
выходные данные
7 5 3 10 7 3

G. Переместить в начало

ограничение по времени на тест: 6 секунд
ограничение по памяти на тест: 512 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

Вам дан массив $a_1 = 1, a_2 = 2, ..., a_n = n$ и последовательность операций: переместить элементы с l_i по r_i в начало массива. Например, для массива $2, 3, 6, 1, 5, 4$, после операции $(2, 4)$ новый порядок будет $3, 6, 1, 2, 5, 4$. А после применения операции $(3, 4)$ порядок элементов в массиве будет $1, 2, 3, 6, 5, 4$.

Выведите порядок элементов в массиве после выполнения всех операций.

Входные данные

В первой строке входного файла указаны числа n и m ($2 \le n \le 100\,000$, $1 \le m \le 100\,000$) — число элементов в массиве и число операций. Следующие m строк содержат операции в виде двух целых чисел: l_i и r_i ($1 \le l_i \le r_i \le n$).

Выходные данные

Выведите n целых чисел — порядок элементов в массиве после применения всех операций.

Пример

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

Условие недоступно на русском языке

I. Эх, дороги

ограничение по времени на тест: 2 секунды
ограничение по памяти на тест: 256 мегабайт
ввод: стандартный ввод
вывод: стандартный вывод

В многострадальном Тридесятом государстве опять готовится дорожная реформа. Впрочем, надо признать, дороги в этом государстве находятся в довольно плачевном состоянии. Так что реформа не повредит. Одна проблема — дорожникам не развернуться, поскольку в стране действует жесткий закон — из каждого города должно вести не более двух дорог. Все дороги в государстве двусторонние, то есть по ним разрешено движение в обоих направлениях (разумеется, разметка отсутствует). В результате реформы некоторые дороги будут строиться, а некоторые другие закрываться на бессрочный ремонт.

Петя работает диспетчером в службе грузоперевозок на дальние расстояния. В связи с предстоящими реформами, ему необходимо оперативно определять оптимальные маршруты между городами в условиях постоянно меняющейся дорожной ситуации. В силу большого количества пробок и сотрудников дорожной полиции в городах, критерием оптимальности маршрута считается количество промежуточных городов, которые необходимо проехать.

Помогите Пете по заданной последовательности сообщений об изменении структуры дорог и запросам об оптимальном способе проезда из одного города в другой, оперативно отвечать на запросы.

Входные данные

В первой строке входного файла заданы числа n — количество городов, m — количество дорог в начале реформы и q — количество сообщений об изменении дорожной структуры и запросов ($1 \le n, m \le 100\,000, q \le 200\,000$). Следующие m строк содержат по два целых числа каждая — пары городов, соединенных дорогами перед реформой. Следующие q строк содержат по три элемента, разделенных пробелами. «+ i j» означает строительство дороги от города i до города j, «- i j» означает закрытие дороги от города i до города j, «? i j» означает запрос об оптимальном пути между городами i и j.

Гарантируется, что в начале и после каждого изменения никакие два города не соединены более чем одной дорогой, и из каждого города выходит не более двух дорог. Никакой город не соединяется дорогой сам с собой.

Выходные данные

На каждый запрос вида «? i j» выведите одно число — минимальное количество промежуточных городов на маршруте из города i в город j. Если проехать из i в j невозможно, выведите -1.

Пример

входные данные
5 4 6 1 2

```
2 3
1 3
4 5
? 1 2
? 1 5
- 2 3
? 2 3
+ 2 4
? 1 5
```

выходные данные

```
0
-1
1
2
```