Содержание

Обратите внимание, входные данные лежат в **стандартном потоке ввода** (он же stdin), вывести ответ нужно в **стандартный поток вывода** (он же stdout).

В некоторых задачах большой ввод и вывод. Пользуйтесь быстрым вводом-выводом.

В некоторых задачах нужен STL, который активно использует динамическую память (set-ы, map-ы) переопределение стандартного аллокатора ускорит вашу программу.

Обратите внимание на `GNU C++` компиляторы с суффиксом `inc`, они позволяют пользоваться дополнительной библиотекой. Под ними можно сдать вот это.

Задачи здорового человека

Задача 1A. Простейшее BST: `upper_bound` [0.3 sec, 256 mb]

В этой задаче вам нужно написать простейшее BST по явному ключу и отвечать им на запросы:

- `+ x` – добавить в дерево `x` (если `x` уже есть, ничего не делать).
- `> x` – вернуть минимальный элемент больше `x` или 0, если таких нет.

Формат входных данных

В каждой строке содержится один запрос.
Все x целые от 1 до 10^9, количество запросов от 1 до 300 000.
Гарантируется, что все x выбраны равномерным распределением.

Формат выходных данных

Для каждого запроса вида «`> x`» выведите в отдельной строке ответ.

Пример

stdin	stdout
+ 1 + 3 + 3 > 1 > 2 > 3 + 2 > 1	3 3 0 2

Подсказка по решению

Случайные данные! Не нужно ничего специально балансировать.

```
struct node {
    node *l, *r;
    int x;
};
```

Замечание

`set/tree` использовать нельзя, все решения с ним будут баниться!

Задача 1B. Двоичное дерево поиска [0.3 sec, 256 mb]

Воспользуйтесь любой стандартной структурой из `C++::STL`.
Или реализуйте сбалансированное двоичное дерево поиска.
Или попытайтесь запихать квадрат.

Формат входных данных

Входной файл содержит описание одной или нескольких операций с деревом.
Операций не больше 10^5. Все числа целые от -10^5 до 10^9.
В каждой строке находится одна из следующих операций:

- `insert` x — добавить в дерево ключ x.
  Если ключ x в дереве уже есть, 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, «`true`», иначе «`false`»
- `next` x — минимальный элемент в дереве, $> x$, или «`none`», если такого нет.
- `prev` x — максимальный элемент в дереве, $< x$, или «`none`», если такого нет.

Формат выходных данных

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

Примеры

stdin	stdout
insert 2 insert 5 insert 3 exists 2 exists 4 next 4 prev 4 delete 5 next 4 prev 4	true false 5 3 none 3

Подсказка по решению

Возможно, вы не привыкли к быстрому вводу выводу. Ознамьтесь с примерами (ссылка в шапке). Ввод в этой задаче примерно такой: `while (!seekEof()) { ... }`

`set`, умный квадрат, сбалансированное bst.

Замечание

`set/tree` использовать можно!

Для искателей острых ощущений

Задача 1C. И снова сумма... [2 sec, 256 mb]

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

- $add(i)$ — добавить в множество S число i, если i там уже есть, S не меняется;
- $sum(l, r)$ — вывести сумму всех элементов x из S: $l \leqslant x \leqslant r$.

Формат входных данных

Исходно множество S пусто. Первая строка входного файла содержит n — количество операций ($1 \leqslant n \leqslant 300\,000$). 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.

Формат выходных данных

Для каждого запроса выведите одно число — ответ на запрос.

Пример

stdin	stdout
6 + 1 + 3 + 3 ? 2 4 + 1 ? 2 4	3 7

Замечание

У этой задаче две версии. В первую зайдёт обычное дерево без перебалансировок, там самое сложное – разобраться, как считать сумму.

В hard-версию зайдёт только AVL/treap/любоенормдерево.