

# Высота дерева

Высотой дерева называется максимальное число вершин дерева в цепочке, начинающейся в корне дерева, заканчивающейся в одном из его листьев, и не содержащей никакую вершину дважды.

Так, высота дерева, состоящего из единственной вершины, равна единице. Высота пустого дерева (да, бывает и такое!) равна нулю.

Дано двоичное дерево поиска. В вершинах этого дерева записаны ключи - целые числа, по абсолютному значению не превышающие  $10^9$ . Для каждой вершины дерева  $V$  выполняется следующее условие:

- все ключи вершин из левого поддерева меньше ключа вершины  $V$ ;
- все ключи вершин из правого поддерева больше ключа вершины  $V$ .

Найдите высоту данного дерева.

Входные данные

Входные данные содержат описание двоичного дерева.

В первой строке находится число  $N$  ( $0 \leq N \leq 200000$ ) - число вершин в дереве.

В последующих  $N$  строках находятся описания вершин дерева.

В  $(i+1)$ -ой строке ( $1 \leq i \leq n$ ) находится описание  $i$ -ой вершины, состоящее из трех чисел  $K_i, L_i, R_i$ , разделенных пробелами - ключа в  $i$ -ой вершине ( $|K_i| \leq 10^9$ ), номера левого ребенка  $i$ -ой вершины ( $i < L_i \leq N$  или  $L_i = 0$ , если левого ребенка нет) и номера правого ребенка  $i$ -ой вершины ( $i < R_i \leq N$  или  $R_i = 0$ , если правого ребенка нет).

Все ключи различны. Гарантируется, что данное дерево является деревом поиска.

Выходные данные

Выведите одно целое число - высоту дерева.

## Проверка корректности

Свойство двоичного дерева поиска можно сформулировать следующим образом: для каждой вершины дерева  $V$  выполняется следующее условие:

- все ключи вершин из левого поддерева меньше ключа вершины  $V$ ;
- все ключи вершин из правого поддерева больше ключа вершины  $V$ .

Дано двоичное дерево. Проверьте, выполняется ли для него свойство двоичного дерева поиска.

Входные данные

В первой строке находится число  $n$  ( $0 \leq n \leq 200000$ ) - число вершин в дереве.

В последующих  $n$  строках находятся описания вершин дерева. В  $(i+1)$ -ой строке файла ( $1 \leq i \leq N$ ) находится описание  $i$ -ой вершины, состоящее из трех чисел  $K_i, L_i, R_i$ , разделенных пробелами - ключа в  $i$ -ой вершине ( $|K_i| \leq 10^9$ ), номера левого ребенка  $i$ -ой вершины ( $i < L_i \leq N$  или  $L_i = 0$ , если левого ребенка нет) и номера правого ребенка  $i$ -ой вершины ( $i < R_i \leq n$  или  $R_i = 0$ , если правого ребенка нет).

Выходные данные

Выведите YES, если данное на входе дерево является двоичным деревом поиска, и NO, если не является.

## Заполнение дерева

Дана структура бинарного дерева. Требуется заполнить её числами от 1 до  $n$  так, чтобы получилось корректное бинарное дерево поиска.

## Входные данные

Входные данные содержат описание двоичного дерева.

В первой строке находится число  $n$  ( $1 \leq n \leq 200000$ ) - число вершин в дереве.

В последующих  $n$  строках находятся описания вершин дерева.

В  $(i+1)$ -й строке ( $1 \leq i \leq n$ ) находится описание  $i$ -й вершины, состоящее из двух чисел  $l_i$  и  $r_i$ , разделённых пробелами - номера левого ребенка  $i$ -й вершины ( $i < l_i \leq n$  или  $l_i = 0$ , если левого ребенка нет) и номера правого ребенка  $i$ -й вершины ( $i < r_i \leq n$  или  $r_i = 0$ , если правого ребенка нет).

## Выходные данные

Выведите  $n$  целых чисел ( $1 \leq k_i \leq n$ ), разделённых пробелами:  $k_i$  - значение в  $i$ -й вершине.

# Простое двоичное дерево поиска

Реализуйте двоичное дерево поиска.

## Входные данные

Описание операций с деревом, количество которых не превышает 100. В каждой строке находится одна из следующих операций:

- insert  $x$  - добавить в дерево ключ  $x$ .
- 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 выходных данных из примера.

## Три друга

Три друга списывают лабораторную работу, каждый из них списывает по  $n$  различных задач. Поскольку друзья не очень умные, они не меняют названия отправляемых на проверку файлов.

По истечении времени, отведенного на написание лабораторной, преподаватель запускает бан-машину и ставит баллы по следующим правилам:

- если задача написана только у одного студента, то этот студент получает 3 балла, поскольку эту задачу он не списывал и не давал списывать;
- если задача списана ровно у двух студентов, то каждый из них получает по 1 утешительному баллу;
- если задача списана всеми тремя студентами, то за нее баллы не начисляются никому.

Выведите финальное количество баллов у каждого студента.

В рамках этой задачи будем считать, что Бан-машина считает решения списанными, если у них полностью совпадают имена файлов.

Входные данные

В первой строке входных данных дается число  $n$  ( $1 \leq n \leq 10000$ ) - количество задач в лабораторной.

Следующие три строки содержат по  $n$  различных слов в каждой — названия файлов с решениями, отправленных каждым из студентов.

Выходные данные

Необходимо вывести 3 числа - количество баллов у первого, второго и третьего студента соответственно.

## Два обхода

Рассмотрим два способа обойти бинарное дерево поиска:

- Вывести значение в текущей вершине и затем рекурсивно запустить процедуру обхода от левого и правого потомка.
- Рекурсивно запустить процедуру обхода от левого и правого потомка и затем вывести значение в текущей вершине.

Требуется по результату обхода первым способом получить результат обхода вторым способом.

Входные данные

В первой строке находится число  $n$  ( $1 \leq n \leq 200000$ ) - число вершин в дереве. Во второй строке содержится результат обхода первым способом:  $n$  различных целых чисел  $k_i$  ( $1 \leq k_i \leq n$ ), разделённых пробелами.

Выходные данные

Выведете перестановку целых чисел  $k_i$ , разделённых пробелами - результат второго обхода.