Задача 1. Обходы дерева

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

По заданной последовательности целых чисел построить дерево двоичного поиска и обойти его в прямом и обратном порядках. Повторяющиеся числа в дерево не вставлять.

Формат входных данных

Во входном файле через пробел записаны целые числа. Количество чисел не превосходит 1000.

Формат выходных данных

В первую строку выходного файла нужно вывести значения, содержащиеся в построенном дереве в прямом порядке обхода, а во вторую — в обратном. Числа выводить через пробел.

Пример

input.txt	output.txt
5 1 10 -3 12 1 9 4	5 1 -3 4 10 9 12 -3 4 1 9 12 10 5

Задача 2. Высота дерева

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

По заданной последовательности целых чисел построить дерево двоичного поиска. Повторяющиеся числа в дерево не вставлять. Необходимо посчитать высоту получившегося дерева.

Формат входных данных

Во входном файле через пробел записаны целые числа. Количество чисел не превосходит 1000.

Формат выходных данных

В выходной файл нужно вывести одно целое число — высоту получившегося дерева.

Пример

input.txt	output.txt
5 1 10 -3 12 1 9 4	2

Задача 3. Количество листьев в дереве

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

По заданной последовательности целых чисел построить дерево двоичного поиска. Повторяющиеся числа в дерево не вставлять. Необходимо посчитать количество вершин, не имеющих сыновей, т.е. листьев дерева.

Формат входных данных

Во входном файле через пробел записаны целые числа. Количество чисел не превосходит 1000.

Формат выходных данных

В выходной файл нужно вывести одно целое число — количество листьев в заданном дереве.

Пример

input.txt	output.txt
5 1 10 -3 12 1 9 4	4

Задача 4. Количество вершин в дереве на заданном уровне

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

По заданной последовательности целых чисел построить дерево двоичного поиска. Повторяющиеся числа в дерево не вставлять. Необходимо посчитать количество вершин, расположенных на заданном уровне получившегося дерева.

Формат входных данных

В первой строке входного файла указан уровень дерева, на котором нужно посчитать количество вершин — это неотрицательное целое число, не превосходящее 1000.

В следующей строке через пробел записаны целые числа. Количество чисел не превосходит 1000.

Формат выходных данных

В выходной файл нужно вывести одно целое число — количество вершин, расположенных на заданном уровне получившегося дерева.

Пример

input.txt	output.txt
1 5 1 10 -3 12 1 9 4	2

Задача 5. Дерево двоичного поиска слов

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

По заданным словам построить дерево двоичного поиска и обойти его в инфиксном порядке. Повторяющиеся слова в дерево не вставлять.

Формат входных данных

Входной файл содержит строки, в каждой из которых записано по одному слову. Длина каждого слова не превосходит 100 символов. Количество слов не превосходит 1000. Пустых строк нет.

В следующей строке через пробел записаны целые числа. Количество чисел не превосходит 1000.

Формат выходных данных

В выходной файл нужно выдать эти слова, упорядоченные в лексикографическом порядке, по одному на строке.

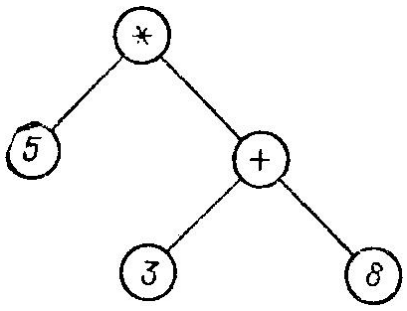
Пример

input.txt	output.txt
orange mallon apple grapes plum banana	apple banana grapes mallon orange plum

Задача 6. Дерево-формула

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

Деревом-формулой называется двоичное дерево, в листьях которого расположены цифры, а в вершинах, которые не являются листьями — знаки арифметических операций $(+, -, *, /)$. На рисунке показано дерево-формула, соответствующее формуле $(5 * (3 + 8))$.



Описать рекурсивную функцию, которая вычисляет (как целое число) значение дерева-формулы.

Формат входных данных

Во входном файле записано выражение в префиксной форме. По этой записи нужно построить двоичное дерево. Длина строки во входном файле не превосходит 1000 знаков.

Формат выходных данных

В выходной файл нужно выдать одно целое число — значение выражения, заданного деревом.

Если возникнет деление на 0, то выдать слово `NO`.

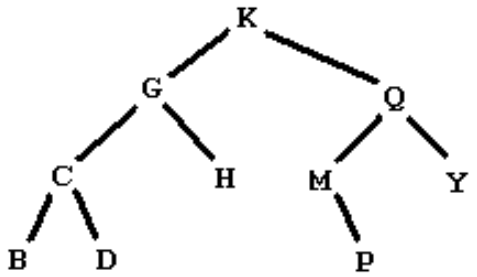
Пример

input.txt	output.txt
*5+38	55

Задача 7. Листопад

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

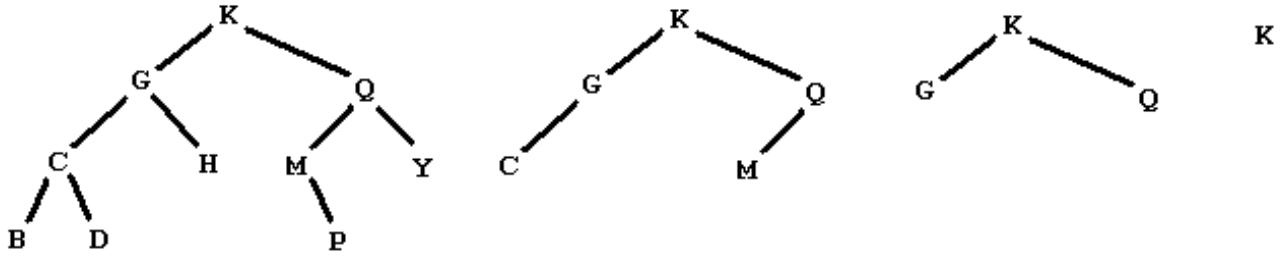
На рисунке изображено графическое представление двоичного дерева поиска символов.



Рассмотрим следующую последовательность действий на дереве двоичного поиска символов:

- Удалить листья и вывести данные удаленных листьев.
- Повторять пока дерево не пусто

Начиная от дерева, нижнего слева, мы получим последовательность показанных деревьев, и затем пустое дерево.



При этом мы последовательно будем удалять следующие данные:

```
BDHPY
CM
GQ
K
```

Ваша задача состоит в том, чтобы исходя из такой последовательности строк листьев дерева бинарного поиска символов, выдать префиксный обход этого дерева.

Формат входных данных

Входной файл состоит последовательности одной или нескольких строк заглавных символов. Строки содержат листья, удаленные из бинарного дерева поиска в последовательности, описанной выше. Символы в строке перечисляются в возрастающем алфавитном порядке. Входной файл не содержит пробелов и пустых строк.

Формат выходных данных

Для входных данных существует единственное бинарное дерево поиска, которое даст последовательность листьев. Выходной файл состоит из строки, состоящей из префиксного обхода такого дерева, без пробелов.

Примеры

input.txt	output.txt
BDHPY CM GQ K	KGCBDHQMPY
AC B	BAC

Задача 8. Сортировка деревом поиска

Источник:	основная*
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 9. 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 ###