Деревья поиска.

Вам предстоит реализовать класс, представляющий из себя бинарное дерево поиска.

A. *Добавление.*
В этой задаче в первой строчке входного файла записывается N — количество элементов, которые будут добавлены в дерево. Затем, в строчку выписаны N чисел, которые нужно добавить. Каждый элемент нашего дерева уникален. То есть, если мы два раза добавим в пустое дерево 57, в дереве будет только один элемент.
В ответ выпишите pre-order обход вашего дерева.

Input	Output
4 1 3 2 4	1 3 2 4

B. *Удаление.*
В первой стоке записано N количество операций с изначально пустым деревом поиска. Далее идут N строчек с описанием операций: слово `insert` или `remove`, далее число. Как и ранее, каждый элемент в дереве уникален, если требуется удалить несуществующий элемент — дерево остается неизменным.
Выведите получившееся дерево в pre-order порядке.

Input	Output
3 insert 1 remove 1 insert 1	1

C. *Поиск интервала.*
В первой строке вводится количество чисел в начальном дереве. Во второй строчке элементы этого дерева поиска. Далее вводится M — количество элементов для поиска. Далее идет строчка из M чисел. Для каждого числа, если оно есть в начальном множестве, выведите само это число. Если же его нет, выведите два числа — наибольшее из меньших и наименьшее из больших. Если одного из них нет, выведите вместо него искомое число.

Input	Output
3 13 666 57 3 -1 13 128	-1 13 13 57 666

D. *Pre oreder.*
Ваня утверждает, что выписал pre-order обход дерева поиска. Проверьте, не ошибся ли он. На вход программе подается длина массива, затем его элементы. Выведите `YES`, если этот массив явлевляется pre-order обходом некоего дерева поиска, выведите `NO` в противном случае.

Input	Output
4 3 1 4 2	NO

E. *Pre-order to post-order*
В первой строке вводится количество элементов в дереве поиска, затем выводится pre-order обход этого дерева. Выпишите post-order обход. Гарантируется, что входной массив правда является корректным pre-order обходом.

Input	Output
3 2 1 3	1 3 2