Задача A. Двоичное дерево поиска (маленькое, 12,5 баллов)

Имя входного файла:	`bst.in`
Имя выходного файла:	`bst.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Реализуйте двоичное дерево поиска.

Формат входного файла

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.
- `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`», иначе «`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.

Все числа во входном файле целые и по модулю не превышают 10^9.

Формат выходного файла

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

Примеры

bst.in	bst.out
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

Задача B. Двоичное дерево поиска (12,5 баллов)

Имя входного файла:	`bst.in`
Имя выходного файла:	`bst.out`
Ограничение по времени:	2 секунды
Ограничение по памяти:	256 мегабайт

Реализуйте сбалансированное двоичное дерево поиска.

Формат входного файла

Входной файл содержит описание операций с деревом, их количество не превышает 100000. Формат операций смотрите в предыдущей задаче.

Формат выходного файла

Выведите последовательно результат выполнения всех операций `exists`, `next`, `prev`.

Задача C. Вирусы и антивирусы (25 баллов)

Имя входного файла:	`virus.in`
Имя выходного файла:	`virus.out`
Ограничение по времени:	3 секунды
Ограничение по памяти:	256 мегабайт

Антивирусная IT-компания имеет официальную иерархическую структуру управления. В ней есть босс — единственный сотрудник, над которым нет начальника. Каждый из остальных сотрудников подчинён ровно одному сотруднику — своему начальнику. Начальник может иметь нескольких подчинённых и отдавать или передавать приказы любому из них. Приказы могут передаваться от одного сотрудника другому только по цепочке, каждый раз от начальника к его подчинённому. Сотрудник А *главнее* сотрудника Б в этой иерархии, если А может отдать или передать приказ сотруднику Б непосредственно, или через цепочку подчинённых. Босс главнее любого сотрудника.

Оказалось, что все сотрудники объединены ещё в одну организованную подобным образом тайную иерархическую структуру, производящую компьютерные вирусы. В тайной структуре может быть другой босс, а у сотрудников — другие начальники.

Будем называть пару сотрудников А и Б *устойчивой,* если А главнее Б и в основной, и в тайной иерархических структурах.

Требуется написать программу, определяющую количество устойчивых пар в компании.

Формат входного файла

В первой строке задано число N — количество сотрудников компании ($1 \leqslant N \leqslant 100\,000$).

Во второй строке — N целых чисел a_i, где $a_i = 0$, если в официальной иерархии сотрудник с номером i является боссом, в противном случае a_i равно номеру непосредственного начальника сотрудника номер i.

В третьей строке — N целых чисел b_i, где $b_i = 0$, если в тайной иерархии сотрудник с номером i является боссом, в противном случае b_i равно номеру непосредственного начальника сотрудника номер i.

Нумерация сотрудников ведётся с единицы в том порядке, в каком они упомянуты во входном файле.

Формат выходного файла

Выходной файл должен содержать единственное число — количество устойчивых пар.

Примеры

virus.in	virus.out
3 0 3 1 0 1 1	2
5 2 0 1 3 4 3 1 0 2 4	7