Lookup Performance

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

A *binary search tree* is a rooted binary tree whose nodes store keys so that each node's key is greater than all the keys in the node's left subtree and less than those in its right subtree.

A binary search tree can be used to maintain sorted sets and allows to perform different types of queries on the set. A query that we are considering in this problem is finding the number of values in the range $[L, R]$ within the set.

Let S be the set of all keys in the binary search tree. You are given two values L and R. The query is to find the number of such $x \in S$ so that $L \leq x \leq R$. The following recursive function computes this value when called with the parameters `lookup(root, L, R)`, where `root` is the root of the binary search tree.

```
function lookup(v, L, R):
  if v == null:
    return 0
  if L <= v.min and v.max <= R:
    return v.count
  if v.max < L or R < v.min:
    return 0
  res = 0
  if L <= v.key and v.key <= R:
    res += 1
  res += lookup(v.left, L, R)
  res += lookup(v.right, L, R)
  return res
```

Values `v.left`, `v.right`, `v.min`, `v.max`, `v.key`, and `v.count` are the fields of the nodes of the binary search tree.

- `v.left` and `v.right` are the left and the right children of node v, respectively.
- `v.min` and `v.max` are the minimum and the maximum keys in the subtree rooted at node v.
- `v.key` is the key of node v.
- `v.count` is the number of nodes in the subtree rooted at node v.

You are given a binary search tree with integer keys. You are also given queries, each query consisting of two integers L and R. Find the number of calls of the `lookup` function that are made when `lookup(root, L, R)` is called, including the initial `lookup(root, L, R)` call itself.

Input

The first line contains an integer n ($1 \leq n \leq 200\,000$) — the number of nodes in the binary search tree.

The next n lines describe the nodes of the tree. The i-th of these lines contains three integers l_i, r_i, and k_i denoting the left child, the right child and the key of the i-th node. If the node does not have the left or/and the right child, the corresponding value is 0 ($l_i = 0$ or $i < l_i \leq n$; $r_i = 0$ or $i < r_i \leq n$; $-10^9 \leq k_i \leq 10^9$). The root of the tree is at node 1 and it is guaranteed to be a well-formed binary search tree.

Note that the values `v.min`, `v.max` and `v.count` are not given in the input explicitly, since they can be deduced from l_i, r_i and k_i.

The next line contains an integer q ($1 \leq q \leq 200\,000$) — the number of the queries.

Each of the next q lines contains two integers L and R ($-10^9 \leq L \leq R \leq 10^9$) — the parameters given to the `lookup` function.

Output

Output q lines, the i-th line containing a single integer — the number of calls of the `lookup` function that are made for the i-th query.

Example

standard input	standard output
7 2 3 4 4 0 2 5 6 8 0 0 1 0 7 5 0 0 9 0 0 6 5 2 7 0 10 2 8 4 4 3 3	7 1 7 3 3