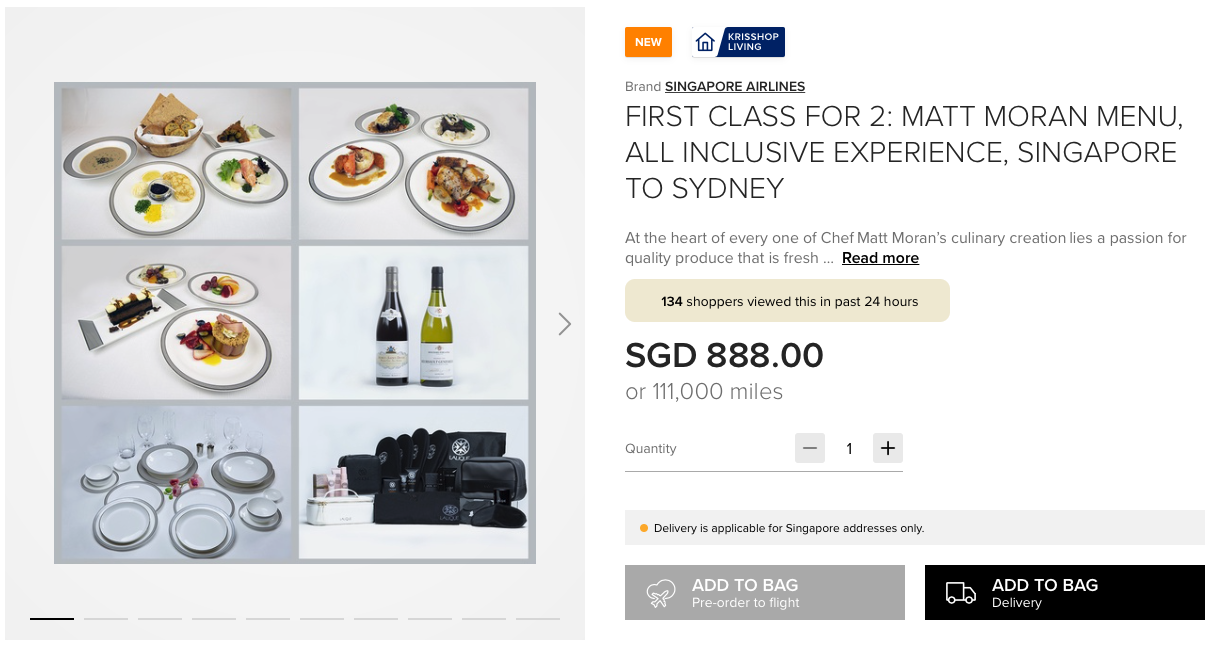
**Uncle Grandpa and LCA**

Now Uncle Grandpa is only 1 problem away from winning the Binary Tree Challenge. The last problem is as follows:

“Given a binary search tree and queries. For each query, you are given 2 nodes, and you need to find their Lowest Common Ancestor(LCA)”

By the time you read this problem, Uncle Grandpa has already won the challenge and take home the prize of 01 set of SIA’s First Class All-Inclusive Experience package worth 888.00 SGD

If you can solve this problem, Uncle Grandpa promises to reward you with nothing! Hey, let’s win it!

**Definition:** A node is defined to be an ancestor of node if lies on the path from the tree root to . By this definition, a node is an ancestor of itself as well. The LCA of 2 nodes and is the lowest (i.e. deepest) node that is the ancestor of both and .

## Input

The first line contains two integers – the number of nodes of the tree and the number of queries

The second line contains integers ( - is the value of the node.

The next lines, each will be in one the following 2 forms:

* means that the node has the node(1-based) as its left child
* means that the node has the node(1-based) as its right child
* The next lines, each will contains two integers represent a query.

**It’s guaranteed that:**

* The given tree will be a binary search tree
* the root of the tree is at node 1.
* (where is the height of the tree)

## Output

For each query, print out on a new line a single integer is the label of the LCA of the two given nodes.

## Examples

|  |  |
| --- | --- |
| Input (lca1.in) | Output (lca1.out) |
| 7 3  10 7 12 5 8 14 20  L 1 2  R 1 3  L 2 4  R 2 5  R 3 6  R 6 7  4 5  6 7  2 3 | 2  6  1 |

## Explanation:

For the 1st query, both node 1 and node 2 are ancestors of both node 4 & node 5. Yet, 2 is lower, so the answer is 2

For the 2nd query, there are 3 nodes that are ancestors of both node 6 & node 7. They are node 1, node 3 and node 6. Among these nodes, node 6 is the lowest, so the answer is 6

For the 3rd query, only node 1 is the ancestor to both node 2 and 3, so the answer is obviously 1.

## Note:

1. A skeleton file has been given to help you. You should not create a new file or rename the file provided. You should develop your program using this skeleton file.
2. You are free to define your own helper methods and classes (or remove existing ones) if it is suitable but you must put all the new classes, if any, in the same skeleton file provided

## Skeleton File

You can find the skeleton file Lca.java in the lab package.

End of the problem statement

------------------------------------------------------------------------------------------------------------------------

**Interesting facts about the problem:** Actually this problem can be solved without the tree being a BST (just a tree is enough) and without the limit of the depth. For those who are interested, you can read about Binary Lifting and Range Minimum Query (RMQ). Also, with the condition that the tree is a BST, there exists a very elegant solution for the problem as well!