Assignment – Binary Tree Recursion

Goal: To gain experience working with recursion in the context of binary trees.

Implement the following methods:

- `nodeCount() -` Returns an integer as to the total count of nodes in the tree.
- `leavsesCount() -` Returns an integer as to the total count of nodes that have zero children.
- `singleChildCount() -` Returns an integer as to the total count of nodes that have exactly 1 child.
- `levelCount() –` Returns the number of nodes on that level of a tree (root node is level 1).

General strategy is to do recursion like the preOrder, inOrder, and postOrder traversal methods, where there will be a public method callable by the application developer, and a private method that keeps the recursion going.

The assignment has two approaches to solving these methods. One uses a persistent counting variable, such as a global variable, by reference, or pointer to a variable. You simply use recursion and count as needed. The second approach uses return values that are int, not void. When a method returns, it should return the accumulation of 1) the count on the left child side, 2) the count on the right child side, and 3) the node it is pointing to, if the particular node applies to the rules for the method. For example

```
int count = nodeCount( /* something to go left*/ );
count += nodeCount( /* something to go right*/ );
count += something for itself
return count;
```

Finally, you must complete the following lambda:

- `myTree.runCustomLambda([](){});`

For this lambda, nothing needs to be captured in the [] region. The parentheses () region should accept a single int by reference. The code {} region is where the parameter int value is doubled.