***Lab Report***

I finished my program's cpp file to get the functions I needed to do the operations and build the tree using the array.

The following functions are included in the cpp file:

In get size, the return keyword is used to determine the size of an array ().

Use get left index to travel from the node's head to the node's left side ().

To retrieve the left node, we use the relation (2\*index) +1.

get it done To get from the node's head to the right side, use node().

We use the relation (2\*index)+2 to get the correct node.

get parent() returns the index of the parent node using the return keyword; if the node selected is -1, it returns -1.

get value() returns the value at the supplied index from the array.

It's called left null() and it checks whether or not the selected node is null.

It returns a bool value after examining the left node.

right null(): This function determines if the selected node is null or not.

It returns a bool value after checking the correct node.

go left(): This function moves the cursor to the left node from the current node.

The go right() function moves the current node to the right.

reset to head(): This function resets the index value to zero.

The Insert() function is used to insert a value into an array.

This function also checks for null values in the array and inserts the value to the left if it is less than the provided value and to the right if it is greater.

I used a for loop in the main function of the source file to add a random value to the array using the add item() method, and then verified the array's size once the loop stopped.

I calculated the time it takes to compile using the following statements.

Before the loop, there is a statement like this: high quality clock::now = auto start ()

After the loop, this statement is used: auto duration = duration castmicroseconds>(stop - start); auto stop = high resolution clock::now();