Lab4 Report

**Introduction:**

In this lab I was given a B-tree and I had to perform many operations using this B-tree. I had to find the minimums and maximums at a particular level of the B-tree, turn the whole B-tree into a sorted list, find the number of nodes on a particular level, print out all of the nodes on a certain level, determine which nodes/leaves were full, and find the depth of a given key. All of these operations required me to traverse the B-tree in order to complete the objective, whether it be finding a depth of a key, or finding a minimum or maximum of a depth.

**Proposed Solutions:**

Turning the tree into a sorted list was an absolute mess at the beginning. The first thing I did was make an if-statement that checks to see if the tree is a leaf or not, if it’s a leaf, I turn the item the tree item in the form of a list. Then I made an empty list to fill it up, I used a for-loop to traverse the tree, I filled up a temporary list using recursion. I then add the temporary list to the empty list along with the specific tree item, tree.item[i], then I had to make a list for the last child, child[-1]. I return everything after empty list and everything in the last child list are added.

For the methods that get the minimum and maximum of a particular level of the tree, all I needed to do was traverse the tree till I got to the desired level, and then return the 0 item if it’s a minimum, or the -1 item it’s the maximum. I got to the desired level by making an if statement checking to see if the depth was zero, if the depth is zero, it’d return the -1 or 0 tree item, if not, a recursive call is made using the first or last child of the tree (depending on if you’re looking for the max or min) and the depth is subtracted by one after each call until the depth is zero. To make sure that the depth actually existed an if-statement was put into place to make sure that the depth wasn’t greater than the height of the tree, if it was a -1 would be returned.

The first step taken to check the number nodes at a depth is to check to see if the depth is 0, I used an if-statement to check whether the node is 0, if it is a 1 is returned because the root node is always going to be 1, then another if-statement is made to check if it’s a leaf. If it is a leaf, a 0 is returned. I used a for-loop to traverse the B-tree and made a counter to count the number of nodes at a level. I used a recursive call to add to the counter, so each recursive call is either going to add a 0 or a 1 to the counter.

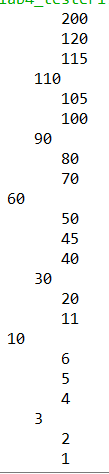
Printing the nodes at a level was pretty easy, I just traversed the B-tree until I got to the desired level and printed the tree items. To determine the number of full nodes in a B-tree, I compared the lengths of the tree items to the max items, if they were the same then that item would be considered a full node, so I assigned a to 1, and if not a was 0. I made a for-loop to traverse the B-tree and used recursion to add the number of full nodes to a, and then a gets returned at the end. Finding the number of full nodes for leaves was practically the same as finding the number of full nodes for the whole B-tree, the only difference is I had to check to see if the item was leaf or not. If it was a leaf, and it was full then a would be 1, but if it wasn’t full or if it was full but not a leaf, a would be 0.

To find the depth of a key the first step I took is to check to see if the key would be at depth 0, if it were 0 would be returned, then I check if it were a leaf or not. If it were a leaf, -1 would be returned, indicating that the key is not in the B-tree. Then I made the for-loop to traverse the tree, and a recursive call was made, but this time in the recursive call I used the find child method, to find the correct child that the key would be in. So, at the end 1 is added to the depth until the key is found. I also made an if-statement that checks to see if the depth is -1 then -1 is returned because the key is not in the B-tree.

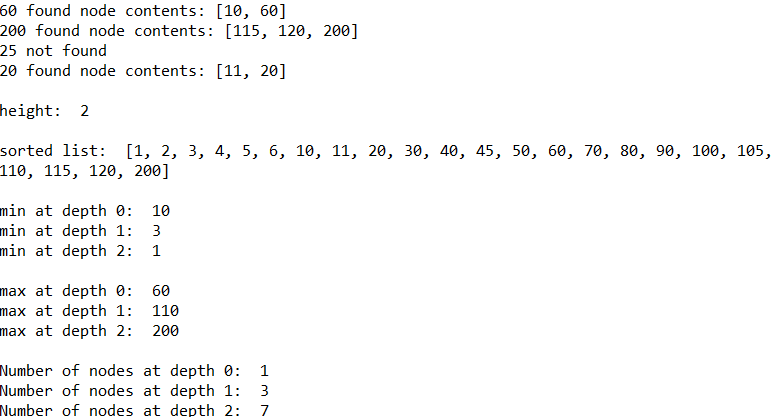
**Experimental results:**

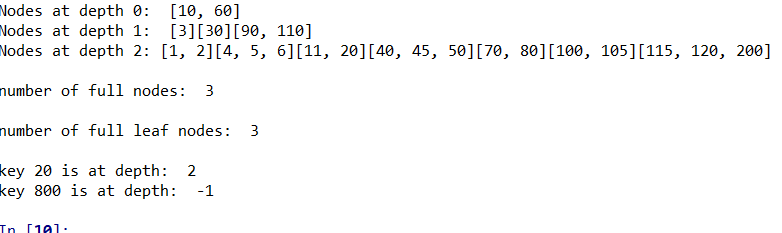
For this lab I used the B-tree that was initially provided. But for the methods that required me to find the full nodes, I had to change the max items to three, instead of five, because on the B-tree provided there was not a single item with the length of five, the longest length was 3. Below is a screenshot of the B-tree that was used, and the results that I got using that B-tree.

**The B-Tree Used**



**The Results Produced**

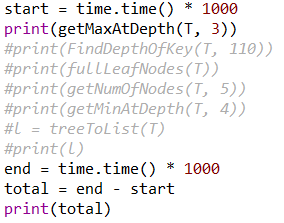




For the runtimes I had to run each method several times to get different times, because most of the times I ran them I kept getting 0.0. I also changed the number of items in the list, I made a list with 150 items in it, then made inserted the items in the list to the B-tree. The most common time I got was 0.0, occasionally I would get a 0.99 or 0.33 after running the method many times. Below is a table that represents the times and methods.

|  |  |  |
| --- | --- | --- |
| treeToList(T) | 0.0 | 0.33 |
| getMinAtDepth(T, 4) | 0.0 | 0.99 |
| getNumNodes(T, 5) | 0.0 | 1.0 |
| FullLeafNode(T) | 0.0 | 0.99 |
| FindDepthOfKey(T, 110) | 0.0 | 0.99 |
| getMaxAtDepth(T, 3) | 0.0 | 0.05 |

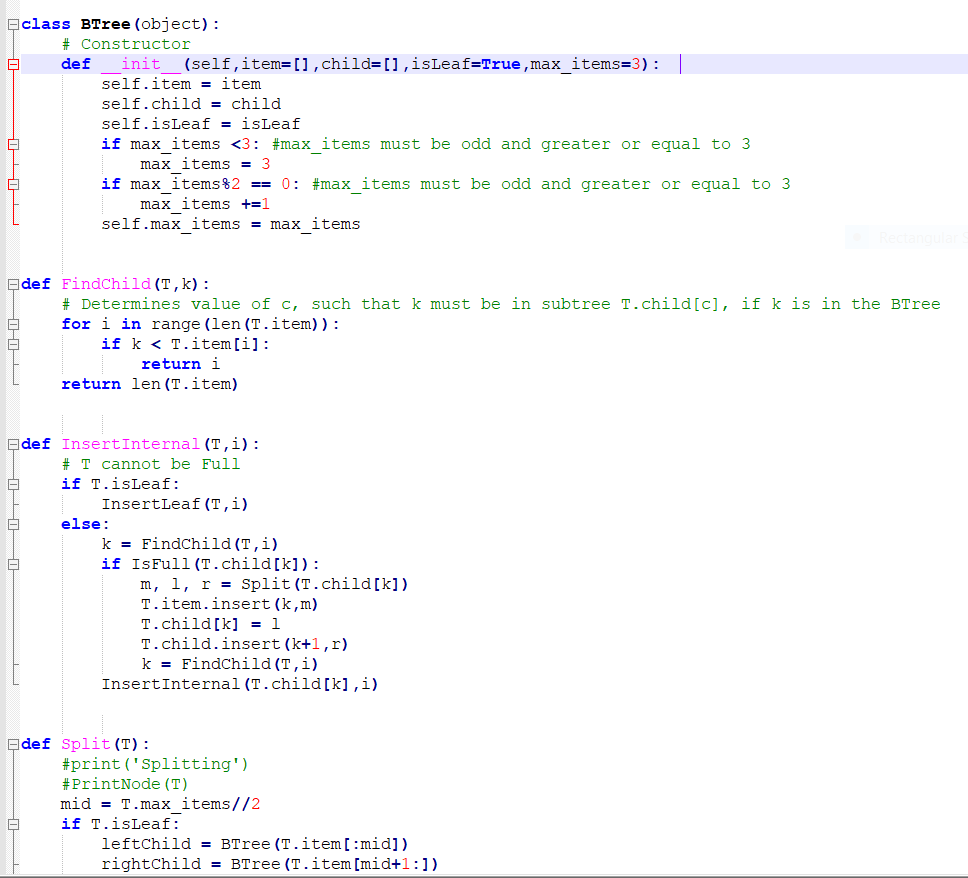
Below is the piece of code that I used to get the times in milliseconds for each method made.

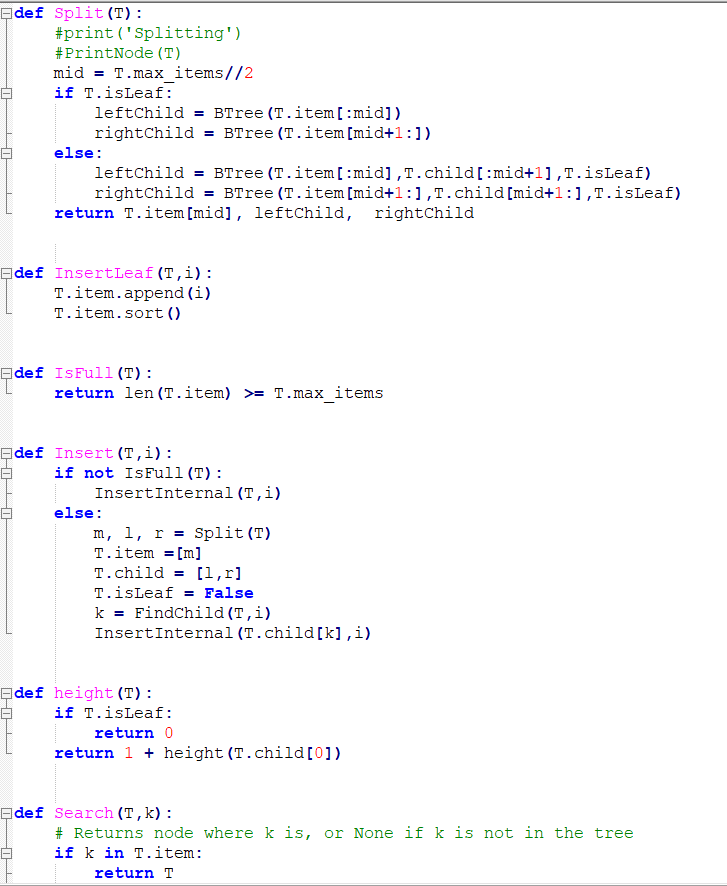


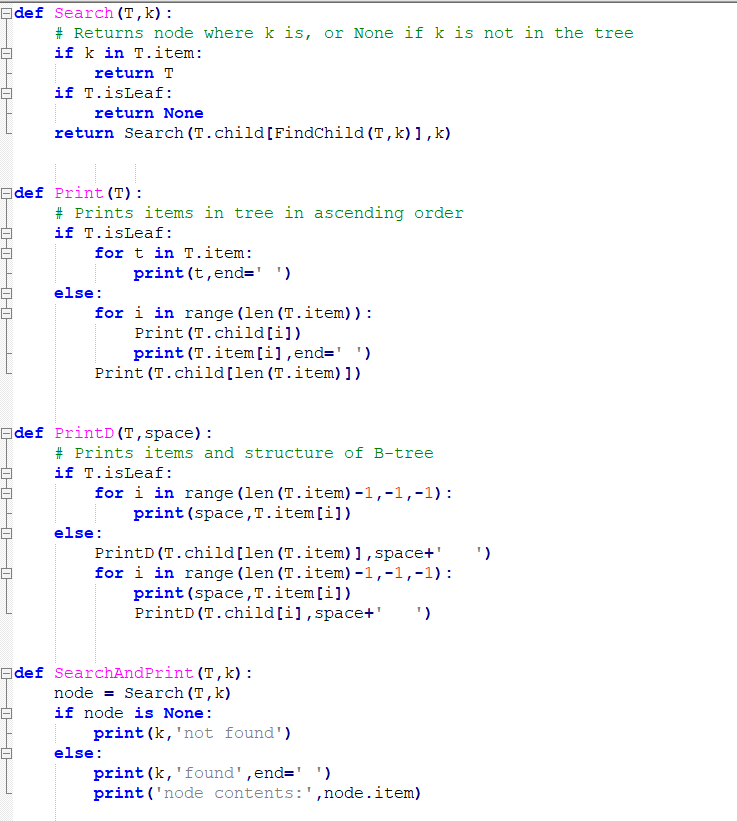
**Conclusion:**

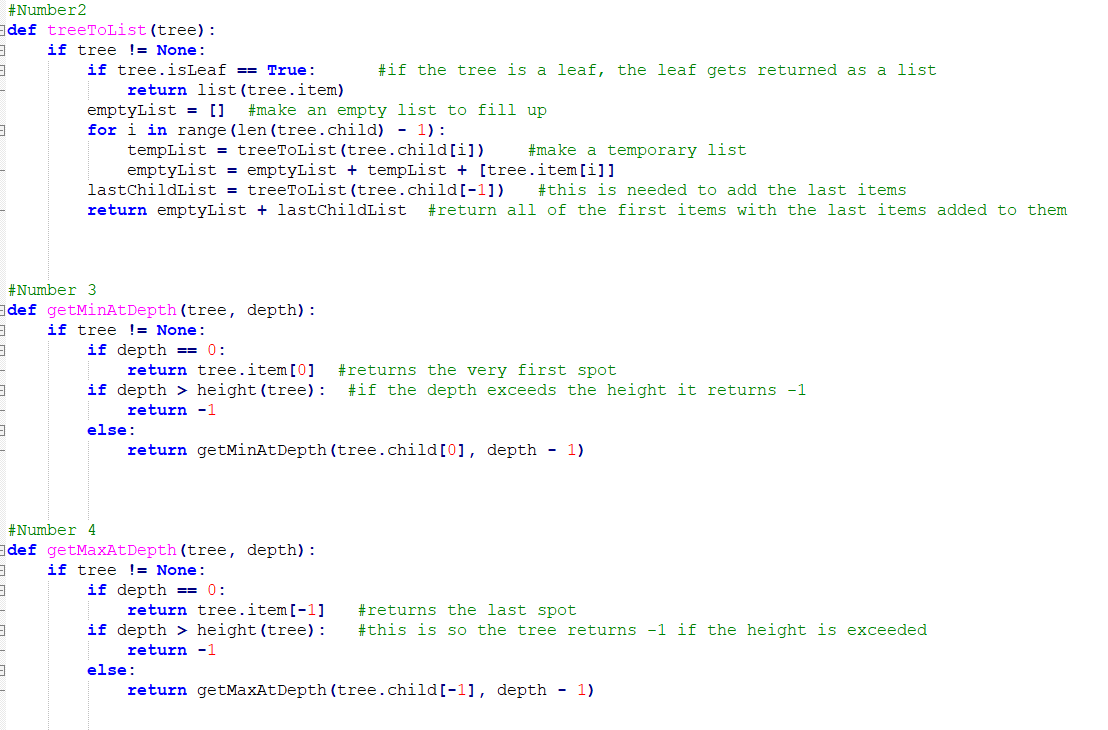
I learned how to traverse a B-tree in this lab. I learned about the child and item list and how they both differ from each other. I learned that the child list is the list that points to another node, and the item list is the list that actually contains the items in the node. I also learned that for-loops and recursive are a necessity to traverse the B-trees.

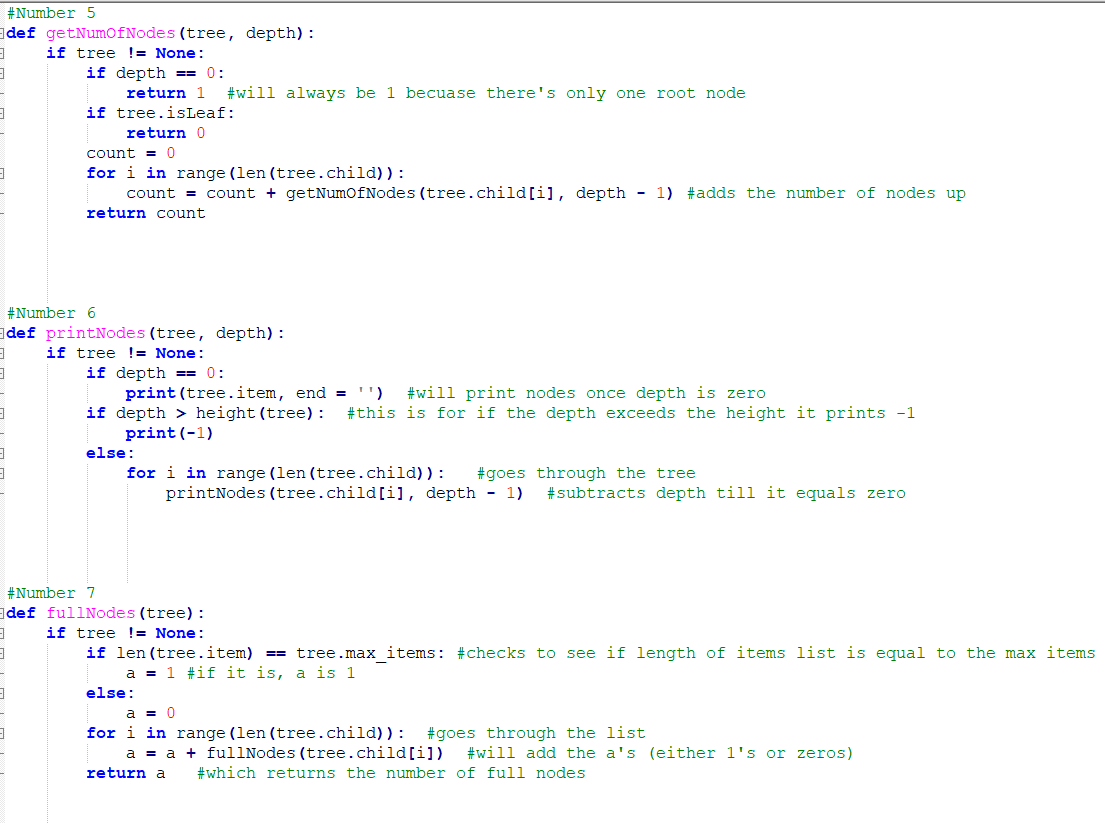
**Appendix:**

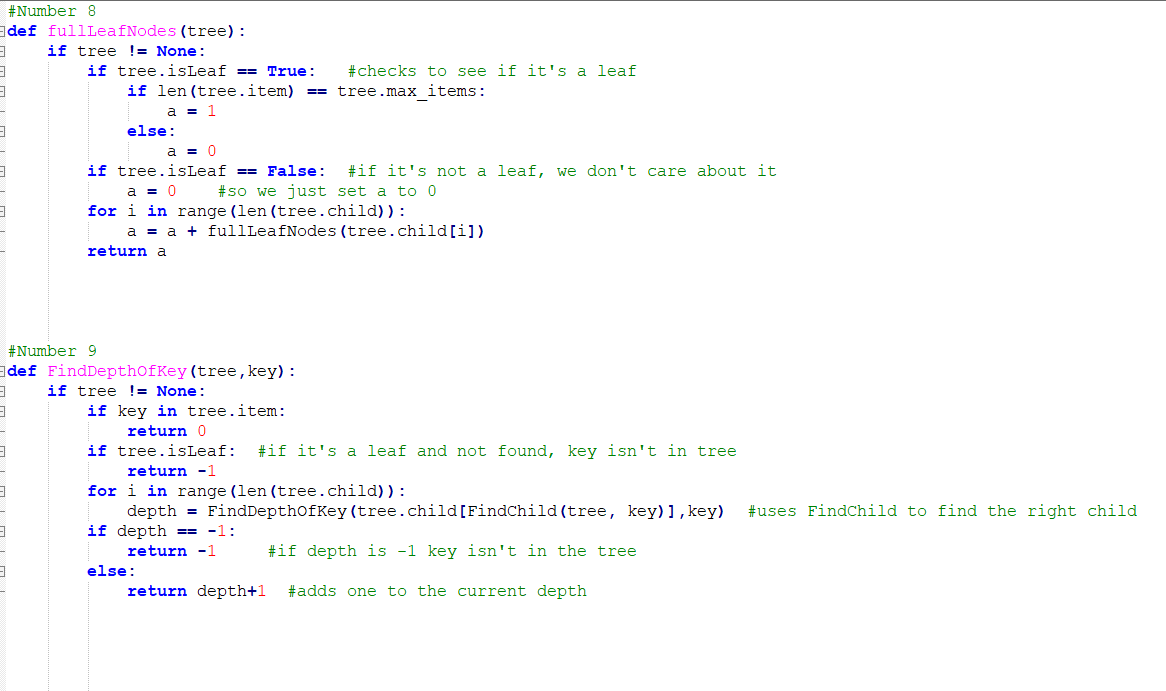


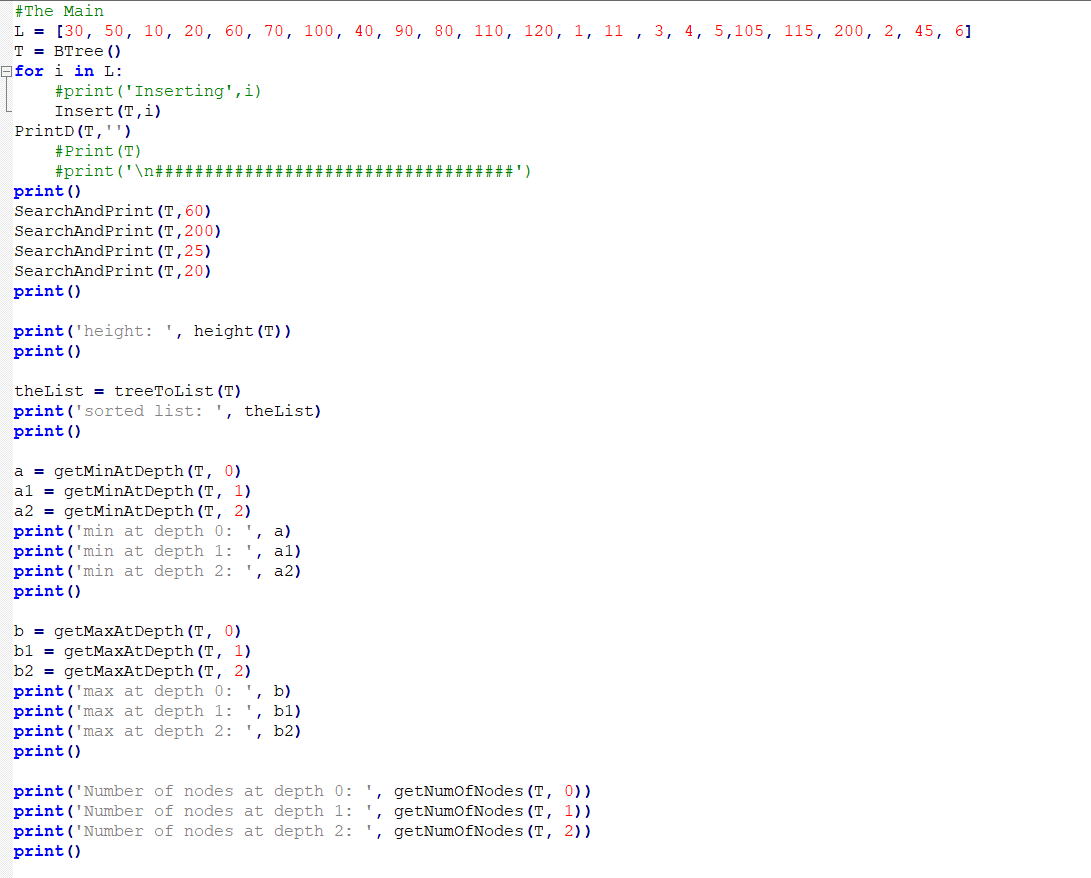


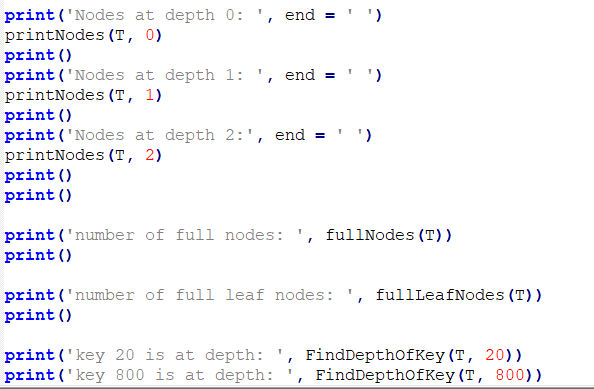












“I certify that this project is entirely my own work. I wrote, debugged, and tested the code being presented, performed the experiments, and wrote the report. I also certify that I did not share my code or report or provided inappropriate assistance to any student in the class.”

Joey Roe