P6 Tree Data Structure

- Due Jun 7, 2018 by 12:20pm
- Points 100
- Available May 29, 2018 at 12am - Jun 15, 2018 at 11:59pm 18 days

This assignment was locked Jun 15, 2018 at 11:59pm.

In this program you will be writing a very simple and incomplete version of the Binary Search Tree data structure. It is incomplete, because we will only be writing 3 functions for the class. Here is the Tree class definition from Tree.h, you will need to write the 3 member functions for the Tree class (insert, op << and print):

```
 // Tree class definition from Tree.h
class Tree {
 public:
 class Node {
  public:
    Node() : left(nullptr), right(nullptr), data(0){}
    Node(int val) : left(nullptr), right(nullptr), data(val) {}
    int data;
    Node *left, *right;
 };

 Tree() : count(0), root(nullptr) {}
 bool insert(int newVal);
 friend ostream& operator << (ostream& str, const Tree& tree);  // calls
print and
                                                                 // outputs
count
private:
 void print(ostream& ostr, Node* curNode) const;  // recursive function
 Node* root;
 int count;
};
```

Requirements:

- The insert member function should use loops to find the correct place for the new data, and the print function must use recursion to print out the sorted data from the Tree.

- Inside the insert function you need to give the following "debug" output:

- *insert in empty tree* when this is the case
- *checking left/right* when moving to the left or right down tree
- *inserting TRUE/FALSE* able to insert or not
  {for an example look at the Program 6 Notes Tree.pdf in the Lecture section on Canvas}

- Insert the following numbers into the tree in this order:

6   9   3   5   5   7   2   10   4   1   8

....and then output the data in sorted order and count from the list

- turn in a paper copy of your main, the output, Tree.h, Tree.cpp, and yes there should be a program header before main and a short class header before the Tree.h