CIS263 Assignment Four

Dr. Denton Bobeldyk

Complete each of the following:

1. Programmatically implement a tree that includes the following:
    a. A node that has 3 properties:
        i. value
        ii. leftChild
        iii. rightChild
    b. Pointer or variable that indicates the root of the tree

2. Create three methods that print out the values of the tree using an inorder, preorder and postorder tree walk.

3. Write-up pseudocode for a method that runs in linear time that determines if the tree passed to it is a binary search tree. The method can only take one argument (or none) and must have a base case. After the pseudocode has been developed, implement it using one of the approved languages.

Hand-in:

1. Two sample trees (hand written or digitally rendered), one sample should satisfy the search tree order property, the other one should not. Each tree must have a minimum of 8 nodes.
2. The pseudocode you created for step 3.
3. The output demonstrating the functionality of your program for each of the above sample trees.
4. A file containing the implementation source code (no zip files).

Approved programming languages: C, C++, C#, Python, Java.

Grading Rubric (scroll down):

Recursive linear-time algorithm Implementation Weight 15.00 %	Percent 0.00 Not completed, or not in linear time	Percent 50.00 Completed, but doesn't run in linear time	Percent 100.00 Complete and runs in linear time
Two sample trees illustrated Weight 15.00 %	Percent 0.00 Neither tree illustrated	Percent 50.00 One tree illustrated	Percent 100.00 Both trees illustrated clearly, one which satisfies, one which does not
Functionality Demonstrated Weight 20.00 %	Percent 0.00 Not demonstrated clearly	Percent 50.00 Limited demonstration	Percent 100.00 Clearly demonstrated
Method follows assignment specifications Weight 30.00 %	Percent 0.00 Does not have a base case and takes more than one argument	Percent 50.00 Does not have a base case or takes more than one argument	Percent 100.00 Has a base case and only takes one argument
Print methods implemented Weight 20.00 %	Percent 0.00 Inorder, preorder, postorder methods not implemented or not implemented properly	Percent 50.00 Inorder, preorder, postorder methods not fully implemented or not all implemented properly	Percent 100.00 Inorder, preorder, postorder methods implemented properly

See blackboard for point breakdown.