CS2400 Fall 2020 Project 3

Total points: 100

Due date: Friday, November 13, 2020

Purpose:

1. Master the Binary Tree structure and its Node representation
2. Understand recursions for tree operations of the BinaryTree class and the BinaryNode class

"Please start working on this assignment as early as possible!"

Task Description:

In this assignment, you will complete the attached java programs by implementing some specific tree methods.

(40 pts) **Task 1: Implement** the following **4** methods related to post-order traversal.

- In "BinaryTree.java"
  - public void postorderTraverse();
    - *Note: this method calls the method postorderTraverse(BinaryNode<T> node) which is a recursive BinaryTree method to perform postorder traversal of the **whole** tree*
  - private void postorderTraverse(BinaryNode<T> node)
    - *Note: this **recursive** method performs postorder traversal of a subtree rooted **at a given node.***
  - public void postorderTraverse_callBinaryNodeMethod()
    - *Note: this method calls the method postorderTraverse_binaryNodeMethod() to perform postorder traversal of the **whole** tree.*
- In "BinaryNode.java"
  - public void postorderTraverse_binaryNodeMethod()
    - *Note: this **recursive** method performs postorder traversal of a subtree rooted **at a BinaryNode object which calls the method**.*

Please do **NOT** call the method *postorderTraverse_binaryNodeMethod()* or *postorderTraverse_callBinaryNodeMethod()* inside the method *postorderTraverse(BinaryNode<T> node)*. You need to implement recursion for the method *postorderTraverse(BinaryNode<T> node)* itself.

(20 pts) **Task 2: Implement** the following **2** methods that return the height of a node or a subtree

- In "BinaryTree.java"
  - public int getHeight_callBinaryNodeMethod()
    - *Note: this method calls the method getHeight_binaryNodeMethod() to return the height of the **whole** tree.*

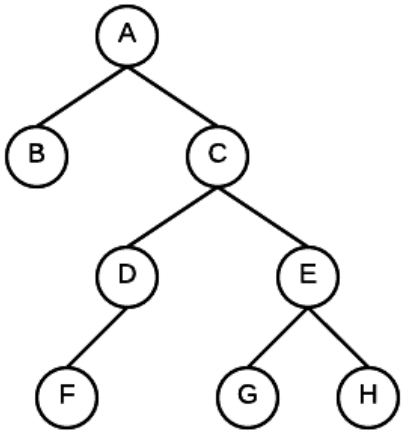
- In “BinaryNode.java”
  - public int getHeight_binaryNodeMethod()
    - *Note: this **recursive** method returns the height of a subtree rooted **at a BinaryNode object which calls the method.***

(20 pts) **Task 3: Implement** the following **2** methods that return the number of nodes of a node or a subtree

- In “BinaryTree.java”
  - public int getNumberOfNodes()
    - *Note: this method calls the method getNumberOfNodes(BinaryNode<T> node) to return the number of nodes of the **whole** tree.*
  - private int getNumberOfNodes(BinaryNode<T> node)
    - *Note: this **recursive** method returns the number of nodes of a subtree rooted **at a given node.***

(20 pts) **Task 4: Implement** the following method to create the 2nd testing example in the client program

- In “DriverBT.java”
  - public static void createTree**2**(BinaryTree<String> tree)
    - *Note: this method hardcodes **the following tree structure** with given value to each node. (Hint: please study the createTree1() method of the client program to see how a tree is created)*



What to Submit?

1. Complete source codes for Tasks 1-4, which are “BinaryNode.java”, “BinaryTree.java”, “BinaryTreeInterface.java”, “DriverBT.java”, “EmptyTreeException.java”, and “TreeInterface.java”.
   Note: Please test your source codes using the **Eclipse** IDE and see if the codes are executable. **Non-executable programs will result in a grade of zero.**
2. Please zip all documents as yourname_p3.zip and submit it in blackboard.
   Note: You will be graded based on the quality of your program.