

“Please start working on this assignment as early as possible!”

CS2400 Project 3

Total points: 100

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

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**.*

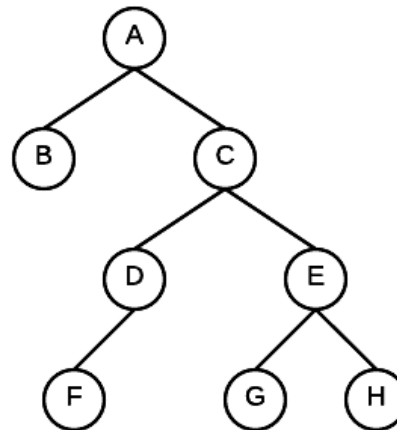
“Please start working on this assignment as early as possible!”

(20 pts) **Task 3: Implement** the following 2 methods that return the number of nodes of a tree 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 createTree2(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”. I expect more than just 2 or 3 commits to GitHub. Please add “2404s21” and “arahman7752” to your organization on GitHub so that we’ll have access to all repos of your organization.
2. Unit testing of the methods written by you.
3. Link to your screencast video (please make it accessible via link) introducing your code and demoing one or more runs.
4. README.md with the video link, “who did what”, and extra features.
5. Please submit a file on Canvas containing a link to your GitHub repo.

Note: Non-executable programs will result in a grade of zero. You will be graded based on the quality of your program.