CS 445: Data Structures
Summer 2016

Assignment 4

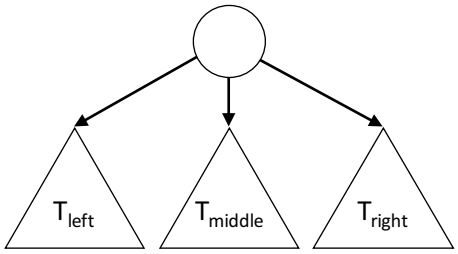
Assigned: Wednesday, July 20 **Due:** Tuesday, August 2 11:59 PM

---

1 Background

In this assignment, you will be implementing a data structure as described in its ADT (represented using a Java interface). Specifically, you are provided with `TernaryTreeInterface`, an interface describing a tree where each node can have up to 3 children. Such a tree is defined as either:

1. Empty; or
2. of the following form, where $\mathsf{T}_{\mathsf{left}}$, $\mathsf{T}_{\mathsf{middle}}$, and $\mathsf{T}_{\mathsf{right}}$ are ternary trees.



When implementing this class, you can use the `BinaryTree` source code provided with the textbook (available at `https://cs.pitt.edu/~bill/445/a/a4binary.zip`) as a starting point, or you can start from scratch. You are also allowed to use classes from the Java Collections Framework such as `java.util.Stack` or `java.util.LinkedList` (these may be useful, e.g., in implementing tree traversal iterators), but you **cannot** use any Java-provided tree structures (e.g., `javax.swing.tree.TreeNode`).

You can download `TernaryTreeInterface` and its superinterfaces at the following link: `https://cs.pitt.edu/~bill/445/a/a4interfaces.zip`. Do not modify these provided files.

2 Tasks

You must implement a class `TernaryTree` that implements the interface `TernaryTreeInterface`. In implementing `TernaryTreeInterface`, you will also need to implement its superinterfaces, `TreeInterface` and `TreeIteratorInterface`.

`TreeInterface` requires basic tree operations such as `getHeight`, `getNumberOfNodes`, etc.

TreeIteratorInterface requires methods that create and return iterators for performing various traversals of the tree. Your TernaryTree class **must include the following inner classes that implement these iterators:** PreorderIterator, PostorderIterator, and LevelOrderIterator. These iterators *do not* need to support the remove() operation (i.e., the remove() method can simply throw java.lang.UnsupportedOperationException like the examples in the book).

Furthermore, **you do not need to implement getInorderIterator.** Instead, this method should also throw a java.lang.UnsupportedOperationException. In addition, **include in the comments** for this method a short description of why TernaryTree does not support inorder traversal.

In addition to the methods required by the interfaces, your class *must* have the following constructors:

- public TernaryTree(): initializes an empty tree
- public TernaryTree(T rootData): initializes a tree whose root node contains rootData
- public TernaryTree(T rootData, TernaryTree<T> leftTree, TernaryTree<T> middleTree, TernaryTree<T> rightTree): initializes a tree whose root node contains rootData and whose child subtrees are leftTree, middleTree, and rightTree, respectively.

You may want to develop a TernaryNode class to represent the nodes of the tree, similar to the BinaryNode class discussed in the textbook and in lecture. As before, you may use the source code provided with the textbook as a starting point.

You are also highly encouraged to write a test client that allows you to test the functionality of your implementation of TernaryTree prior to submission. You will be graded on each method's functionality as it compares to the descriptions in the interfaces.

3 Submission

Create a zip file containing *only* java files (no .class files). Include the provided interfaces as well as your TernaryTree.java file, and any other required files. Your TA should be able to unzip your submission, add the test client for grading, then compile and run from the command line without additional changes. Be sure to test this procedure (unzip, compile, and run a client if you include one) before submitting your zip file.

If you use an IDE to develop your ADT, export the java files from the IDE and test that they still compile on the command line. Do not submit the IDE's project files.

In addition to your code, you may wish to include a README.txt file that describes features of your program that are not working as expected to assist the TA in grading the portions that work as expected.

Submit your zip file according to the instructions at https://cs.pitt.edu/~bill/445/#submission

Your project is due at 11:59 PM on Tuesday, August 2. Be sure to test the submission procedure in advance of this deadline: no late assignments will be accepted.