Institute of Technology, University of Washington Tacoma
TCSS 342 Data Structures, Winter 2019
Assignment 3 – Trees
Value: 10% of the course grade
Due: Friday, 15 February 2019, 23:59:59

Purpose:

This assignment has a number of purposes. It is intended to provide practice with the following:

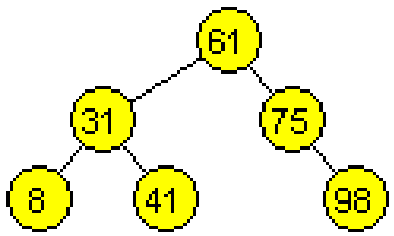
- Implement methods on tree data structures (particularly recursive methods)
- unit testing, coding to conventions, proper code documentation

Assignment Description:

You will implement and test several methods for binary trees. The starter code for this project includes interfaces and classes described in chapter 10 of the course textbook. You will make changes or additions to the LinkedBinaryTree class and the BinaryTreeADT interface as explained below. Do not make changes to other parts of the starter code.

1. Implement methods of the LinkedBinaryTree class which were left as exercises in the textbook. (NOTE: Reading chapter 10 carefully (particularly section 10.7) will help.)
   Do NOT add any new fields to the class (such as a mySize field).
   Methods you must implement:
   - `contains()`
   - `getRootNode()`
   - `getRootElement`
   - `iteratorPreorder()` and `iteratorPostorder()` (Hint - use `iteratorInorder()` as an example)
   - `preOrder()` and `postOrder()` (Hint - use `inOrder()` as an example)
   - `toString()` – use an in order iterator
   - `size()` – (Hint - traverse the tree and count the nodes)
   - `getHeight()` – think recursively. What is a useful base case? What is the recursive case?
2. Add the following method to the BinaryTreeADT interface and implement this method in the LinkedBinaryTree class:

```
int countLeafNodes()
// This method returns the number of nodes in the tree which have 2 null children
// If this method were called on the tree shown below it should return 3.
// (8, 41, and 98 are leaf nodes)
```



NOTE: Your implementation for this method **must be recursive.** Do NOT change the signature of this method. The method listed above will probably not be recursive itself. Instead, I suggest that you have the method call a recursive helper method to do the work.

3. Run the provided PostfixTesterMain program which uses the methods you have developed. Run multiple test cases until you are satisfied that your method implementations are correct. Optionally also write unit tests for your methods and place those unit tests in the tests package in the project.

Getting Started:

Create your Eclipse project by downloading the `hw3-project.zip` file from the Assignment 3 page on Canvas, importing it into your workspace. Use "Refactor" to change "USERNAME" in the project name to your UWNetID.

NOTE: The starter code produces one compile error which will be resolved when you add the countLeafNodes() method (see step 2 above).
The Eclipse plugin tools generate a few warnings on the starter code:

- Checkstyle – 6 warnings
- PMD – 14 warnings

You do not need to modify the starter code to silence these warnings. Some of the warnings cannot be silenced. You are only responsible for the warnings produced by your own code. Keep such warnings to a minimum. If you do not understand a warning then ask about it.

Submission and Grading:

When you have committed the final revision of your code to SVN, make a note of its Subversion revision number. To get the revision number, *perform an update* on the top level of your project; the revision number will then be displayed next to the project name.

Submit (on Canvas) an *executive summary*, containing the Subversion revision number of your submission, an "assignment overview" (1 paragraph, up to about 250 words) explaining what you understand to be the purpose and scope of the assignment, and a "technical impression" section (1-2 paragraphs, about 200-500 words) describing your experiences while carrying out the assignment. The assignment overview shows how well you understand the assignment. The technical impression section allows you to describe which aspects of the project that you found difficult or interesting and to give credit for any help you have received.

The filename for your executive summary must be "`username-assignment3.txt`", where `username` is your UWNetID. An executive summary template is available on the Canvas. Executive summaries will *only* be accepted in plain text format – other file formats (RTF, Microsoft Word, Acrobat PDF, Apple Pages) are *not* acceptable.

Internal correctness includes meaningful and systematically assigned identifier names, proper encapsulation, avoidance of redundancy, good choices of data representation, the use of javadoc on all class members, the use of non-javadoc implementation comments on particularly complex code sections, and adherence to coding conventions as identified by the output of the Eclipse plugin tools.

For this assignment, the percentage breakdown is 10% executive summary, 55% external correctness as determined by running my automated tests on your code, 25% internal correctness, and 10% for your unit tests.