# BinaryTree Completion
**Due: Day of lab at 11:59PM**

## 1 BinaryTree Completion Specification

### 1.1 Lab Instructions

- This is an *individual* lab.
- Make sure to read through all of the specifications so your submission is complete.
- Follow all the submission steps in the Setup document by the lab deadline.

### 1.2 Lab Link

The skeleton code for the lab is available at `https://classroom.github.com/a/BnzmxzRw`.

### 1.3 Implementation

This lab introduces a `BinaryTree` class to implement a binary tree. The class is the same as we saw during the lecture, with one addition.

I have provided the class code in the **edu.wit.cs.comp2000** package. You will complete the one additional method – `toPostorderString`. A description of the method's expected behavior is included in the comments.

For your implementation, consider how `toString` formats the output text, and base your solution on that. If you write any additional methods, mark their visibility as `private`.

### 1.4 Bugs

There is a bug in **one** of the existing methods in **BinaryTree.java**. You will have to fix the method in order for all of the operations to work correctly.

The easiest way to identify the method is to write comprehensive unit tests. Once you have identified the method, consider what steps you would like it to take to accomplish its task. Change the current code to perform those steps.

### 1.5 Testing

In addition to the `BinaryTree` code, JUnit tests are provided in the **edu.wit.cs.comp2000.tests** package. You can run these tests to see if the `BinaryTree` implementation is performing correctly. The tests that I have provided check that some of the operations are behaving as expected.

For the remaining two test methods (the ones that fail as not implemented), implement tests that check if those methods work with your binary tree implementation. Make sure that you actually call the method that you are testing, and test for a range of results that you might expect.

### 1.6 Considerations For Each Method

Be considerate in testing your ADT. Your goal is to test all of the possible cases for each `BinaryTree` method. Each test method may have several assert method calls depending on how many possibilities you test for.

## 2 Double Check:

- Have you implemented the `toPostorderString` method?
- Have you written two JUnit tests?
- Have you corrected a buggy method?
- Have you committed/pushed your code from the two files?

## 3 Grading

Each of the 4 **TODO** sections is worth $\frac{1}{4}$ of the lab grade.

Grades and any comments for the lab will be posted to your project on github. Grades will also be posted to Brightspace, eventually.