AADS – Lab Week 2 BST with rotations

Task 1:

Basically, complete the exercise we discussed in the lecture (I've reproduced the text here but you should refer to the document for diagrams.)

As an in-class exercise, we will attempt to establish what operations are required for a left rotation. Probably the simplest way is to use the tree in the diagram with nodes 31, 50, 57, 69, 90, 99 for analysis.

After that you should be able to turn it into code:

1. Write a method called `public void rotateLeft()` for our BinarySearchTree.
   Note: this is just a test method that will rotate the whole tree. Later we can modify it to rotate a given sub-tree.

2. The code is only a few lines long. Hint: you may need to use temporary reference variables (in case, as you perform operations, you overwrite some references that you will need access to later.)

3. How could you then verify - from your test program - that the tree had indeed rotated? Do you think, for example, that printing out an in-order traversal will achieve this?

4. After you have successfully implemented and verified the method, provide a `public void rotateRight()` method.

Task 2:

Change the internal structure of your `rotateLeft()`, `rotateRight()` methods so that they instead call a method, e.g. `rotateSubTreeLeft(root)`. You will experience an issue here. Let's see what you make of it ☺

http://tommikaikkonen.github.io/rbtree/ with accompanying video at https://www.youtube.com/watch?v=g9SaX0yeneU

The following resource should make sense within the context of the previous two links.

http://www.geeksforgeeks.org/red-black-tree-set-2-insert/