CS2123 Data Structures
Assignment 4 - Trees

Compiling your code:

Setup your files such that all of the tree library .java files are in a folder called "binarytrees". Your Driver.java file should be in the level above that folder. To compile your code:

(1) ensure your are in the folder containing your "Driver.java" file.
(2) run the command "javac ./binarytrees/*.java" to compile the tree library
(3) run the command "javac *.java" to compile your Driver file
(4) run the command "java Driver" to run your Driver file

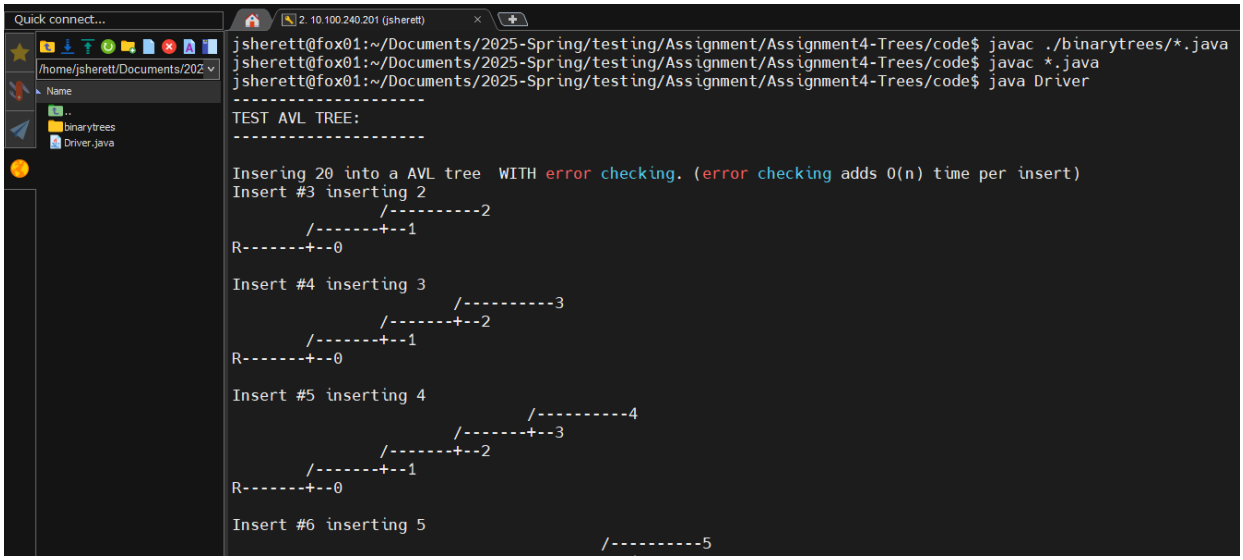

Fig. 1. Compile your files as shown above

AVL Tree - rebalanceTree

Complete the method "rebalanceTree" in the file "AVLTree.java". This method modifies the tree to ensure the balance of every node above x is -1, 0, or 1. Here is a brief outline of the algorithm for rebalancing the tree using AVL trees:

Reminder: the balance of x is the height of the left subtree of x - height of the right subtree of x.

(1) While x is not NULL
  1 if the balance of x is ≤ -2 or ≥ 2
    (i) Set z equal to the child of x with the greater height
    (ii) if the balance of x and the balance of z have different signs
      (A) if the sign of the balance of z is + right rotate on z
      (B) else the balance is $-$ so you left rotate on z
    (iii) if the balance of x is ≥ 2 right rotate on x
    (iv) else the balance is ≤ -2 so you left rotate on x
  2 Set x equal to the parent of x

Each node with balance outside the range -1 to 1 will be reported as an error. If there are no errors the testing code will output "No errors detected! Well done!". Here is the sample output:

```
---------------------
TEST AVL TREE:
---------------------

Insering 20 into a AVL tree  WITH error checking. (error checking adds O(n) time per insert)
Time to insert 20 numbers: 0.00007 seconds.
No errors detected!  Well done!

Insering 100000 into a AVL tree WITHOUT error checking.
Time to insert 100000 numbers: 0.03400 seconds.
```

Your run times may vary a bit depending on what computer you use to run your code. Without tree balancing the 2nd test will take roughly a minute. After implementing tree balancing, the test should run in under a second.

Deliverables:

Your solution should be submitted as "AVLTree.java". Upload this file to Canvas under Assignment 4. **Do not zip your file**.

To receive full credit, your code must compile and execute.