Assignment 4
Binary Search Trees

Release Time	Due Date
March 20, 2017	**April 2, 2017**

Objectives

- Experience manipulating binary search trees.
- Continue to gain more experience with using heaps.
- Practice developing high-performance solutions.
- Analyze the advantages and disadvantages of the binary search trees

Problem Specification

Write a JAVA application to solve the following problem,

1) Generate 100 random integers within the range [1,20].

2) Then insert them into a binary search tree T using an explicit representation. For this assignment we don't need to worry about minimizing the height of the search tree.

3) Each node of T contains two values, one is the key value and the other is the counter representing #repeats of the key value in the randomly generated sequence in step1, in addition to the tree pointers (leftChild, rightChild, parent), sizeOfSubtree and subtreeHeight

   For example: if key 10 gets repeated 3 times, then the node corresponding to key value 10 is:

10	3
keyVal	repeatCount

   If we insert one more 10 into this tree, then there is no extra node added, but this node 's counter will change from 3 to 4. If we delete one 10, then the counter will decrease by 1. In this kind of data structure, a node is deleted only if the counter reaches 0..

4) Print the node whose counter has the largest value, and the node whose counter has the smallest value, and delete these two nodes one after another without destroying the structure of the binary search tree.

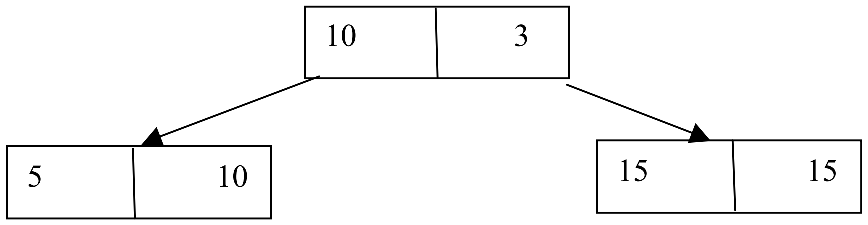
   [***Note:*** *For an efficient implementation, to be able to easily find these nodes after any insert or delete operation, one can maintain a separate min-max heap that's based on the counter values to determine the key values of these nodes (another alternative is to maintain two heaps, one max-heap and another min-heap). To simplify the assignment, we are not requiring this.*

However, for extra 30 points, you may want to implement heap on counter values along with search tree organized on keys. If you decide to do so, clearly identify at the beginning of your code.]

5) Compute and print sum(keyVal) and sum(repeatCount), i.e., additions of all the keys and counters, respectively.

6) Determine and print a distinct keyVal and its repeatCount so that if we insert a node with these values to the tree, the sum(keyVal) equals sum(repeatCount). Insert this node in the binary search tree.

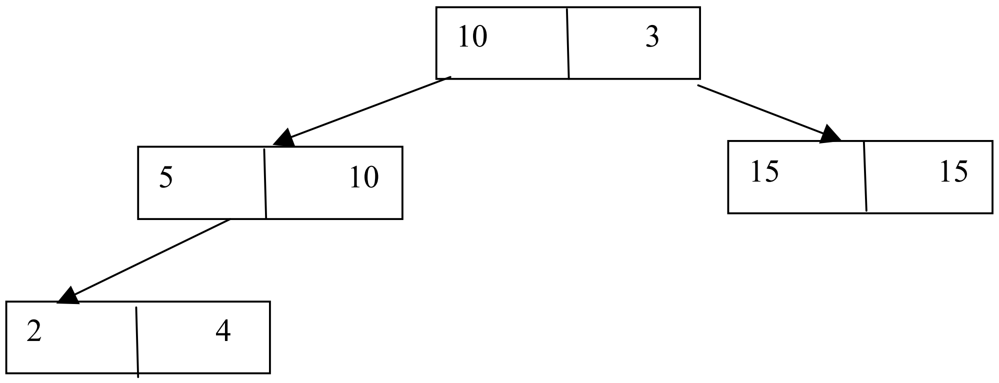
   Example:



Sum(keyVal)= 10+5+15 =30 Sum(repeatCount)=3+10+15=28

Then, as an example, we can insert, say, key 2 four times.

Then the resulting tree is:



Sum(keyVal)= 2+10+5+15 =32 Sum(repeatCount)=3+10+15+4=32

and we have sum(keyVal)=sum(repeatCount).

7) For k = 5, 20, 30 and 70, find and print kth-smallest key in the binary search tree.

8) Repeat steps 1-7 twenty times and print the minimum and maximum heights of the search trees obtained in these 20 experiments.

9) Now traverse the final tree as preorder, postorder, and inorder and print your traversal results. Print so that the values of each node are shown as tuples (keyVal, repeatCount, sizeOfSubtree,

subtreeHeight) and tuples of different nodes are separated by commas. For example for the tree above, your inorder traversal output may look like:

```
(10, 3, 4, 2), (5, 10, 2, 1), (2, 4, 1, 0), (15, 15, 1, 0)
```

Design Requirements

Code Documentation

For this assignment, you must include documentation for your code as generated by JavaDoc. You should have JavaDoc comments for every class, constructor, and method. By default, JavaDoc should output html documentation to a subfolder within your project (/dist/javadoc). Make sure this folder is included when you zip your files for submission. You do not need to submit a hard copy of this documentation.

Hint: http://stackoverflow.com/questions/4468669/how-to-generate-javadoc-html-in-eclipse

Coding Standards

You must adhere to all conventions in the CS 3310 Java coding standard. This includes the use of white spaces for readability and the use of comments to explain the meaning of various methods and attributes. Be sure to follow the conventions for naming classes, variables, method parameters and methods.

Testing

Make sure you test your application with several different values, to make sure it works.

Assignment Submission

- Generate a .zip file that contains all your files, including:
    - Source code files
    - Including any input or output files
- Javadocs
- Readme file – how to compile and run your program and whether you have implemented extra credit part.