1. Write a method called count that is part of the BinaryTree<E> class that returns the number of nodes in the tree (without returning numElt or numberOfElements; calculate it by going through the tree). You'll want to write a countRecursive helper method that does most of the work.

//the countrecursive helper method

public int counthelper() {

return countRecursive(this.root, 0);

}

private int countRecursive(Node<E> pos, int count) {

if(pos == null) {return 0;}

count++;

return count + countRecursive(pos.left, 0) + countRecursive(pos.right, 0);

}

2. Write a method called countSmaller(E x) that is part of the BinarySearchTree<E> class that returns the number of values stored in the tree that are smaller than x (according to compareTo). For example, if the tree stored the letters A, C, G, J, K, Q, Z, countSmaller(H) would return 3, as would countSmaller(G) and countSmaller(I). Note that x may or may not be a value in the tree; your code should work either way. Again you'll want a countSmallerRecursive method that does most of the work. Your method must do the minimum amount of work possible. Hint: The countRecursive method you wrote for part 1 might be useful here.

public int countSmaller(E data) {

return countSmallerRecursive(data, this.root, 0) - 1;

}

private int countSmallerRecursive(E value, Node<E> pos, int count) {

if(pos == null) {

return 1;

} //isn’t this the same amount of work? Worst case is O(n)

if(value.compareTo(pos.data) > 0) {

count++;

int left = countSmallerRecursive(value, pos.left, count);

int right = countSmallerRecursive(value, pos.right, count);

return left + right;

} else {

int left = countSmallerRecursive(value, pos.left, count);

return left;

}

}

3. Draw the structure that would be formed by inserting the words in this question into a minheap. Use lowercase letters.

[Note on question 3 that is not part of the question: use alphabetical ordering to decide what word is smaller than which other word. Words should be added one at a time]

a



be draw



by letters in formed



inserting lowercase the words structure this question into



the minheap use that would

4. Show the new minheap formed by taking your answer to question 3 and doing three successive "remove minimum" operations on it.

draw



inserting formed



lowercase letters in into



minheap that the words structure this question use



the would

5. Suppose a small hashtable has 5 slots in it. Index 0 stores 24, index 1 stores 6, index 2 stores 20, and index 4 stores 14. (Index 3 has nothing stored there.) Assuming that the hashing function is just taking the number mod 5, give a sequence of insertions starting with an empty table that could have led to the table filled as described above. Which numbers caused collisions in your sequence?

Start with 14 (goes to 4)

Then 6 (goes to one)

Then 24 (goes to 0, by the collision with 14)

Then 20 (goes to 2 by collision with 24)

24 and 20 caused collisions in this sequence.