Name: _____ Andrew Id: _____

15-121 Assessment 4

60 minutes. No calculators, no notes, no books, and no computers.
Place answers on the answer sheet.

1. **Short Answer**

   Answer the following in 1 sentence; longer answers will be marked incorrect.

   (a) (2 points) Why might we want to use a Comparator instead of implementing Comparable in a class?

   (b) (2 points) Which order of inserting data into a binary search tree yields its most unbalanced (worst-case) shape?

   (c) (2 points) What is the main difference between a heap and a binary search tree?

2. (4 points) **Binary Search Tree Construction**

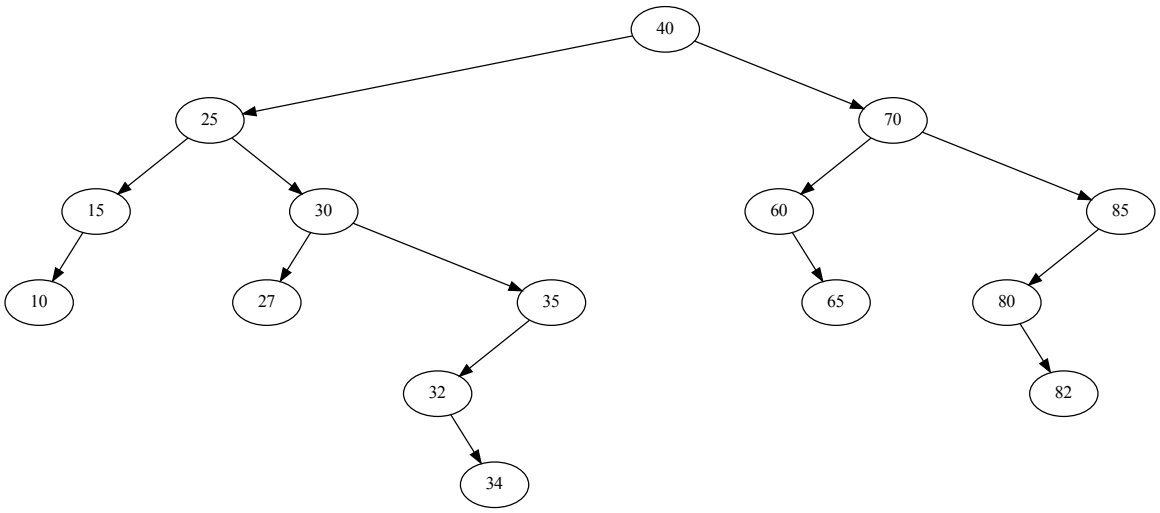
   Imagine you are constructing a binary search tree of integers, and the following integers are added in the following order:

   ```
   45, 28, 67, 15, 33, 89, 71, 19
   ```

   Draw the resulting binary search tree.

3. (3 points) **Binary Search Tree Removal**

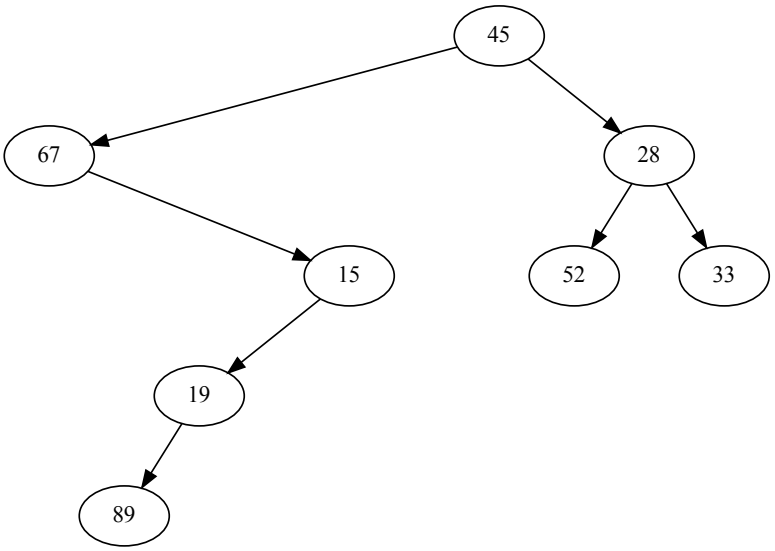
   Consider the following binary search tree:



   Assuming you are using the in-order successor removal technique discussed in class, draw the state of the tree from after 30 has been removed from it.

4. **Binary Tree Traversal**

Consider the following binary tree. Note that it is *not* a binary search tree, it is simply nodes stored arbitrarily in a binary tree.



(a) (3 points) Assuming the tree is traversed using an *in-order* traversal and the nodes printed, what is the resulting sequence? (Assume that left is followed before right.)

(b) (3 points) Assuming the tree is traversed using a *pre-order* traversal and the nodes printed, what is the resulting sequence? (Assume that left is followed before right.)

(c) (3 points) Assuming the tree is traversed using a *post-order* traversal and the nodes printed, what is the resulting sequence? (Assume that left is followed before right.)

5. **Binary Heap**

Consider the following array representation of a binary heap:

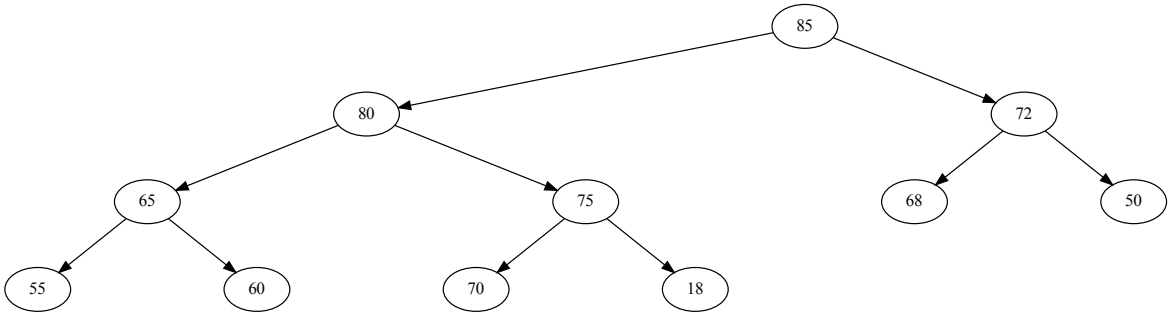
95	90	82	85	78	70	65	80	75	60				

(a) (1 point) Draw the heap represented by this array.

(b) (3 points) Give the array after adding 93 to the heap. Write your final answer in the boxes.

6. **More Binary Heaps**

(a) (1 point) Consider the following heap:



Write this heap in array form. Write your final answer in the boxes.

(b) (3 points) Building on your answer from part a, give the array after calling `removeMax()` on the heap. Write your final answer in the boxes.

7. **Binary Search Tree Free Response**

Consider the following code for a binary search tree of integers.

```
public class BinarySearchTree {
    private TreeNode root;

    private class TreeNode {
        private int data;
        private TreeNode left;
        private TreeNode right;

        private TreeNode(int data) {
            this.data = data;
        }
    }

    public BinarySearchTree() {
        this.root = null;
    }

    public void add(int item) {
        this.root = add(root, item);
    }

    private TreeNode add(TreeNode node, int item) {
        if (node == null) {
            return new TreeNode(item);
        }

        if (item < node.data) {
            node.left = add(node.left, item);
        } else {
            node.right = add(node.right, item);
        }

        return node;
    }
}
```

(a) (6 points) Write the a new method in this class called `allNonNegative()` that returns `true` if all nodes in the tree contain positive integers, and `false` otherwise. An empty tree should return true.

You may write additional helper methods, but you may not modify existing methods or add any new instance variables to `BinarySearchTree`. You also may not use any other data structures, such as an array or an `ArrayList`.

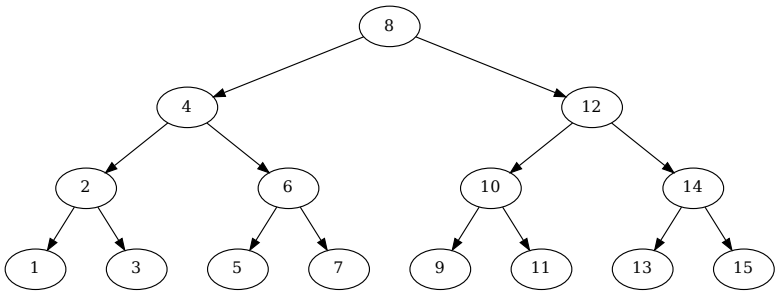
Your method should be as efficient as possible. For example, if the most efficient implementation of this method is $O(\log N)$, then your method should not be $O(N)$. (But if the most efficient implemention is $O(N)$, then of course your method being $O(N)$ is fine.)

(b) (6 points) Write a new method in this class called `getDepth(int value)`, which returns the depth of the node containing the given value. The root is at depth 0, its children are at depth 1, and so on. If the value is not found in the tree, return -1.

You may write additional helper methods, but you may not modify existing methods or add any new instance variables to `BinarySearchTree`. You also may not use any other data structures, such as an array or an `ArrayList`.

Your method should be as efficient as possible. For example, if the most efficient implementation of this method is $O(\log N)$, then your method should not be $O(N)$. (But if the most efficient implemention is $O(N)$, then of course your method being $O(N)$ is fine.)

(c) (8 points) Write the a new method in this class called `sumOddLevels()` that returns the sum of all values stored in the tree that are located on odd levels. The root node is considered to be at level 0. An empty tree should return 0. Consider the following example:



In this case, `sumOddLevels()` should return 80. (Which is $4+12+1+3+5+7+9+11+13+15$)

You may write additional helper methods, but you may not modify existing methods or add any new instance variables to `BinarySearchTree`. You also may not use any other data structures, such as an array or an `ArrayList`.

Your method should be as efficient as possible. For example, if the most efficient implementation of this method is $O(\log N)$, then your method should not be $O(N)$. (But if the most efficient implemention is $O(N)$, then of course your method being $O(N)$ is fine.)