Name: ______________________________ Andrew Id: ______________________

# 15-121 Fall 2021 Quiz 10

Up to 25 minutes. Show your work. No calculators, no notes, no books, no computers, no other people.

1. (5 points) **Binary Search Tree Construction**

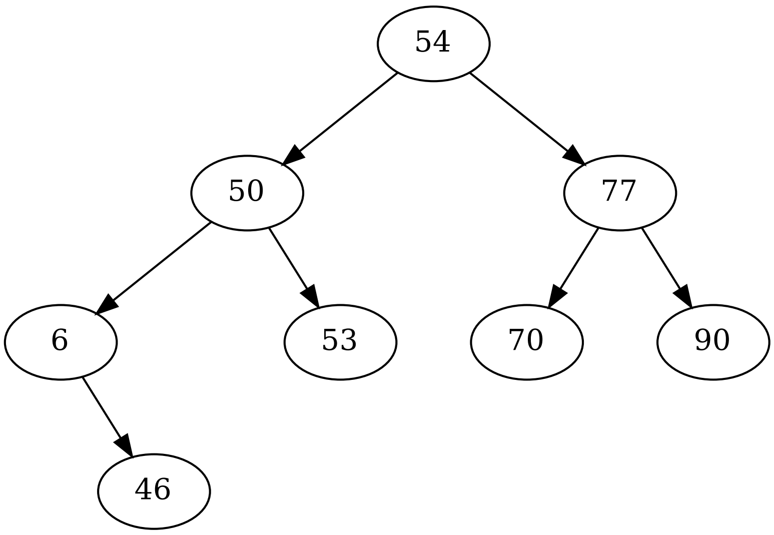
   Imagine you are constructing a binary search tree of integers, and the following integers are added in the following order:

   ```
   16, 67, 85, 55, 11, 43, 71, 87, 3, 44
   ```

   Draw the resulting binary search tree.

2. **Binary Search Tree Traversal**

Consider the following tree:



(a) (1 point) Assuming the tree is traversed *in-order* and the nodes printed, what is the resulting sequence? (Assume that left is followed before right.)

(b) (1 point) Assuming the tree is traversed *pre-order* and the nodes printed, what is the resulting sequence? (Assume that left is followed before right.)

(c) (1 point) Assuming the tree is traversed *post-order* and the nodes printed, what is the resulting sequence? (Assume that left is followed before right.)

3. (6 points) **Binary Search Tree Free Response**

Consider the following code for a binary search tree of Strings. (There is nothing special here, the code is provided just in case you forgot how a binary tree is built.)

```java
public class BinarySearchTree {
    private TreeNode root;

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

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

    public BinarySearchTree() {
        root = null;
    }

    public void add(String item) {
        root = add(root, item);
    }

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

        int res = item.compareTo(node.data);
        if (res < 0) {
            node.left = add(node.left, item);
        } else {
            node.right = add(node.right, item);
        }

        return node;
    }
}
```

The rest of the question is on the next page.

Write the code for a new method in this class called `longest()`, which returns the length of the longest string in the tree. If the tree is empty, return `-1`.

4. (6 points) **Loyalty**

   Imagine you are writing a program that stores information about employees of a company. You write a `Employee` class that has the following information about an employee:

   - First name (`String`)
   - Last name (`String`)
   - Year they joined the company (`int`)

   You choose the natural ordering for employees as being based on their last name, then first name in the case of a tie.

   While writing other parts of the program, you want to sort an `ArrayList` of employees by how many years they have been with the company (in ascending order, meaning that newer employees are listed first), and if two employees have been with the company the same number of years, then by last name, and if that is the same, then by first name. You write the following line of code:

   ```
   Collections.sort(listOfEmployees, new EmployeeSortingThing());
   ```

   Write the class `EmployeeSortingThing` that makes this work as described. (To simplify things, you may just assume appropriately named public instance variables and getter methods exist for all of the attributes of a `Employee`.)