Name: _____

Please submit this homework by following the homework submission instructions on the class Web site. Reminder: You are welcome, and encouraged, to use any resources (e.g., Web sites) to help you with your work. However, *all such help must be clearly noted* in your submissions. Further, no matter what you use, *you must be able to explain* how and why it works. Refer to the class policy for details, and ask for clarifications if you are unsure if something is allowed. Questions marked with $\star$ are optional and may be answered for extra credit.

1. (1 pt.) Write your name in the space provided above.
2. (1 pt.) After completing the programming assignment and uploading your jar file to the FTP server, write the size of your jar file, in bytes, here: _____
3. (36 pts.) Read the textbook section that describes the *nearest common ancestor (NCA)* problem and answer the following questions briefly based on that material.[1]

   (a) What is the difference between an offline problem and an online one?

   (b) What type of traversal does the NCA algorithm perform?

[1] Mark Allen Weiss, *Data Structures and Problem Solving Using Java*, 3rd edition (Addison-Wesley, 2006), Section 24.2.3, pp. 839–842.

(c) True or false: (Justify your answer.) In the state suggested by Figure 24.9, all recursive calls to shaded nodes have finished.[2].

(d) What is the difference between *marking* and *visiting* a node?

(e) Define the *anchor* of a visited node.

(f) Indicate the anchor of each of the labeled nodes (A, B, C, D, p, q, and r) in Figure 24.9.

(g) True or false? (Justify your answer.) The number of equivalence classes (as used by the algorithm) is equal to the length of the access path.

[2] *Idem*, p. 840.

(h) In Figure 24.10, let D's two children be E and F, and let C's other child be G. Indicate the anchors of C, D, E, F, and G just after the recursive call from C returns.

(i) Consider a tree as above, in which leaves represent teams and each interior node represents a game between its two children. Let a and b be two leaves of this tree. Provide an interpretation for the nearest common ancestor of a and b. (What does it represent, in terms of teams and games?)

4. (62 pts.) Write a program that takes as input a tree T and a list of requests $r_1, r_2, \ldots, r_k$, where each request $r_i = (x_i, y_i)$ is a pair of nodes of T, and produces as output a list $a_1, a_2, \ldots, a_k$, where a_i is the nearest common ancestor of x_i and y_i.

   Your code should implement the simple `NCAncestor` interface outlined by Figure 1, which uses the `Tree` and `TreeNode` interfaces from the previous assignment.

   Use the NCA algorithm described in the textbook.[3] An implementation of a union-find data structure, which is used by this algorithm, will be provided for your use. For further details, refer to the class newsgroup and Web site. Also, see Question 5.

5. (5 pts.) $\star$ Instead of using the available implementation of a union-find data structure, use your own union-find implementation. Submit your union-find implementation with the rest of your code, being sure to include the necessary details in the README file.

6. (15 pts.) $\star$ Describe an algorithm for computing the nearest common ancestors that works for directed acyclic graphs[4] in addition to trees. Explain why your algorithm is

[3] *Idem*, Figure 24.11, p. 842.
[4] *Idem*, p. 473.

```
public interface NCA {
   /**
      @return a list of tree nodes that are the nearest common
         ancestors of the corresponding pair of tree nodes in the
         request list.
      @param t the input tree.
      @param requests the request list, consisting of a list of pairs
         of tree nodes.
      @see interfaces Tree and TreeNode.
   */
   public List<TreeNode> nca(Tree t, List<List<TreeNode>> requests);
}
```

Figure 1: An interface for computing nearest common ancestors.

correct, and analyze its asymptotic running time. (You may attach additional sheets of paper as needed to answer this question.) Implement your algorithm and submit it with the rest of your code.