Homework 6
CS5046

Due by 5:30PM, April 14, 2004
Submit by email to murali@cs.vt.edu

This homework will explore the data structures we have been studying in class. Read the material from Chapters 3–6 of Goodrich and Tamassia to help you with this homework. You no longer have to use BlueJ to prepare homeworks. You can use the Java packages that come with the Goodrich and Tamassia book, which you can download from http://net.datastructures.net.

1. (30 points) Implement a class called Reverser whose function is to reverse a collection of Objects. The class has a method called reverse() that has a Collection as a parameter; the signature of the method is public void reverse(Collection c). The method reverses the elements in the Collection c, i.e., when reverse() finishes, c stores its original elements but in reverse order. You should not use the toArray() method in the Collection interface when you implement reverse(). You can use one auxiliary data structure in reverse() but it cannot be an array or an instance of ArrayList. Keep in mind that reverse() can only invoke those methods on c that are specified in the Collections interface. Implement a public static void main(String[]) method in the Reverser class that accepts a list of strings as input from the command-line and uses the Reverser class to compute and print out the reverse of the input list.

2. (30 points) In the net.datastructures package, the LinkedBinaryTree class implements the BinaryTree interface. The BinaryTree interface is a sub-interface of the Tree interface that represents trees where each internal node has exactly two children. Add a public method with the signature int depth(Position node) to the LinkedBinaryTree class. This method should compute the depth of node in the tree. You are not allowed to compute the depth by calling the depth() method recursively.

3. (40 points) Implement a public method with the signature void depths() that computes the depth of every node in the LinkedBinaryTree. The method should print out the object stored at each node and the depth of that node.

4. (Extra credit, 50 points) Solve Problem 3 so that the depths() method runs in \(O(n)\) time, where \(n\) is the number of nodes in the tree.

• Make sure you test all your solutions thoroughly. Consider all types of inputs a user may provide and how you will deal with them. Read Chapter 6 of Barnes and Kölling to refresh your memory about the types of tests you should perform.

• Create a folder called <YourName>-Homework6 and use a separate folder inside this folder for the source code for each problem.

• Submit your homework by emailing the files containing the source code to me. I prefer it if you can zip the entire folder and send the zipped folder to me.