CS 260
Programming Assignment 6

This lab is worth a total of 100 points; 10 points are the self-evaluation, 60 points for the basic lab, and 30 points for the advanced lab.

Base Lab Specification

The first step is to build a class representing a binary search tree for integers. This class should be named Tree. You can base your code off the pseudo code in the binary parse tree document in Moodle.

This lab should follow the course coding requirements and be split into multiple files as per your chosen language. There is a driver provided for this lab. You should test each part of your code as it is developed.

The base methods required are:

- insertValue(value) – add a new node containing value to the tree
- findValue(value) – return true if there is a node containing value, false otherwise
- removeValue(value) – if there is a node in the tree containing value, remove it and return true. If there is not such a node, return false. You should *mark it as removed, not actually remove* the node.
- preOrder() – return a string containing a prefix listing of the tree's contents. *If a node was deleted, add a D* to its value. This method must be coded recursively.
- inOrder() – return a string containing an infix listing of the tree's contents. *If a node was deleted, add a D* to its value. This method must be coded recursively.
- postOrder() – return a string containing a postfix listing of the tree's contents. *If a node was deleted, add a D* to its value. This method must be coded recursively.
- destructor() – required for C++, must be coded recursively.

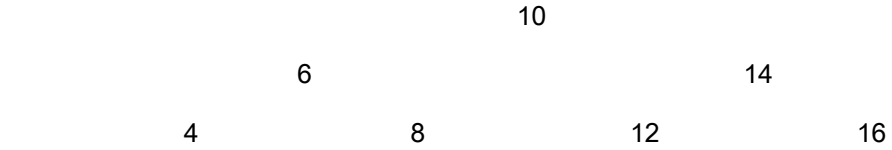
Advanced Lab Specification

For the advanced lab, start with your base lab class and add the two methods to your class. These are not described in the documentation and you will need to think about how a binary search tree is organized and searched to find a way to solve the problem.

- findLarger(value) – search for a node containing value and return value if such a node is found. If no such node is found, return the next larger value in the tree. If there are no larger values in the tree, return -1.
- removeLarger(value) – similar to findLarger but removes the node before returning.

Note this requires that the tree only contains positive integers so -1 is not a valid value.

Example of using findLarger



findLarger(6) – returns 6
findLarger(9) – returns 10
findLarger(12) – returns 12
findLarger(13) – returns 14
findLarger(17) – returns -1