CS 014 Assignment #3

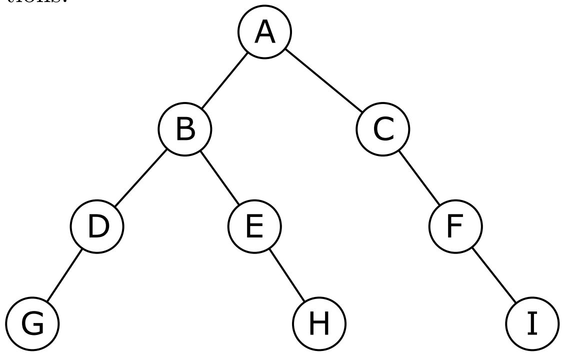
Due on Tuesday 11/07/2017 at the *beginning* of the class

Instructions: Please include the following on the cover page of your assignment:

- Full name
- Student ID
- Your lab section number
- The name of the TA of your lab

Answer the following questions.

1. (10 points) Inspect the tree shown below and answer the following questions.



   (a) (1 points) What is the height of the tree?

   (b) (1 points) What is the height of the subtree rooted at F?

   (c) (1 points) How many siblings does the node E have?

   (d) (1 points) List down the ancestors of the node H in a descending order of their depth.

   (e) (2 points) Enumerate the tree nodes in the preorder traversal. You do not need to show any intermediate step. Only the final order is required.

   (f) (4 points) Repeat the last part for the inorder and the postorder traversals.

2. (10 points) Write down *recursive* tree algorithms for the following problems. In each problem, the input is a pointer to the root of a binary tree. The definition of the tree node is shown below. Your algorithm should support the special case of an empty tree which is denoted by a `NULL` pointer to its root. Analyze the worst-case running time of each algorithm using the Big-Oh notation.

```
class Node {
  int value;
  Node *left, *right;
};
```

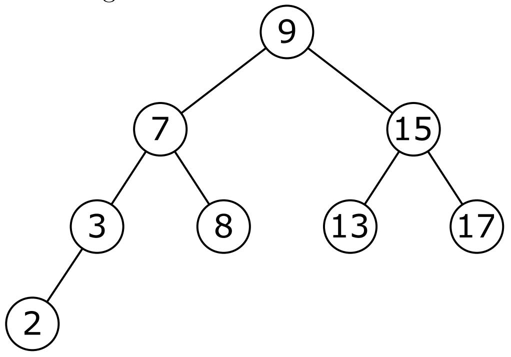
(a) (2 points) Compute the height of the tree. Assume the height of an empty tree is -1.

(b) (2 points) Compute the sum of all values in the tree nodes. The sum of an empty tree is 0.

(c) (2 points) Search for a given integer value `x` and return true if the tree has a node with the given value; false otherwise.

(d) (4 points) Compute the number of nodes at each level of the tree. The return value is a list of integers where the entry at the position $i \in [0, h]$ (h is the height of the tree) indicates the number of nodes with depth i. For an empty tree, an empty list is returned. For example, for the tree used in Question 1, the return value is $[1, 2, 3, 3]$.

3. (5 points) For the binary search tree (BST) shown below, carry out the operations shown below in the given order. The state of the tree after each operation carry out to the next operation. For example, after you insert the value 10 in part (a), assume that the value stays there for all the following parts. In all operations, assume we use the basic insert or delete algorithms of a BST unless otherwise noted.



(a) (1 point) Insert the value 10 in the tree.

(b) (1 point) Insert the value 1 in the tree.

(c) (1 point) According to the AVL tree balancing condition, is the tree currently balanced? List down all the unbalanced tree nodes?

(d) (1 point) Make the tree balanced by making the appropriate rotation on the deepest unbalanced node.

(e) (1 point) Delete the value 9 from the tree. You do not need to balance the tree after deletion.

Note: This assignment should be done individually. You can either deliver it on iLearn or hand it out at the *beginning* of the class. You can either handwrite it or type it on your favorite word processor and submit it as a PDF file. As an acknowledgment for your typing effort, you will get an extra 10% for typing it *neatly* without exceeding 100% of the final grade.