CS 112 – Data Structures

Summer, 2015

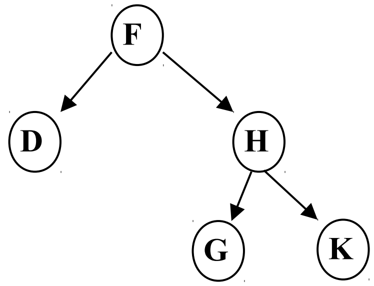
Midterm Exam 2

- **Do not open** this exam until everyone has an exam and the instructor tells you to begin.
- Once you start you will have 100 minutes (1 hour and 40 minutes).
- There are 5 pages in this exam, including this one. Make sure you have them all.
- This exam is closed book – closed notes.
- You must put your cellphone, PDA, Ipod, or other electronic devices in a backpack, etc, and leave it out of your reach. The only exception is that you can use a watch that only has time-related functions (e.g. not a smart watch).
- Write clearly – if we can't read or can't find your answer, your answer is wrong.
- Make clear what is your answer versus intermediate work.
- Please do not sit next to anyone whose name you know

For graders use only:

1		/20
2		/20
3		/20
4		/20
5		/20
6		/20
7		/20
8		/20
9		/20
total		/180

1. For the tree to the right



   a. What is its root? _____ Its leaves ? _______

   b. What is its height?   _________

   c. What is the depth of node D? _________

   d. What nodes are in the subtree rooted at H? _______

   e. Assume the letter written in a node is the data at that node, and assume letters are to be ordered alphabetically. Is this tree a valid Binary Search Tree? Why or why not?

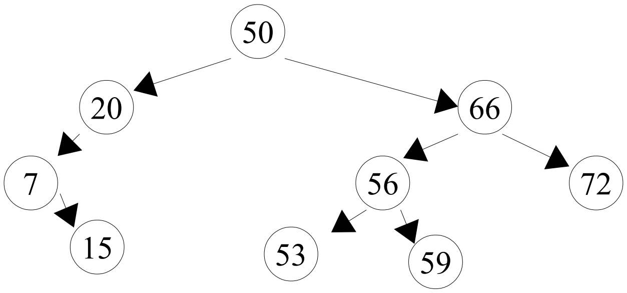
2. Finish the following method copyTree. copyTree makes a copy of the whole tree of TreeNodes rooted at node, and returns the root node of this copy. The original tree is not changed. Hint: Use recursion.

```
public class TreeNode{
   public TreeNode leftSubtree, rightSubtree;
   int data;
   public TreeNode(int dat, TreeNode lst, TreeNode rst){
      data = dat;  leftSubtree = lst; rightSubtree = rst;
   }

   public static TreeNode copyTree (TreeNode node){
       if (node == null){

           return ______________________________________________;
      }else{





               }}
```

3. Boolean method inOrder(BST bstA, BST bstB), where a BST is a binary search tree, should return true if the largest data in bstA is smaller than the smallest data in bstB. If either tree has no nodes in it, inOrder should return false. Finish the boolean method below.

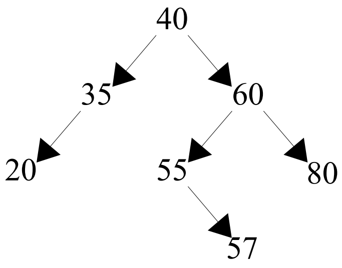
```
public class BSTNode{
   int data; BSTNode lst; BSTNode rst;
}
public class BST{
    BSTNode root;

    public static boolean inOrder(BST bstA, BST bstB){





}}
```

4. Suppose we do the following operations on the Binary Search Tree below. Mark the resulting changes on the tree:
insert 75
delete 66
delete 20



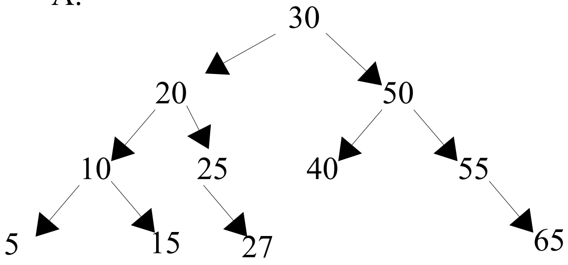
5. Suppose you were searching the Binary Search Tree below for the target 51.

   Circle the numbers would you compare with 51.

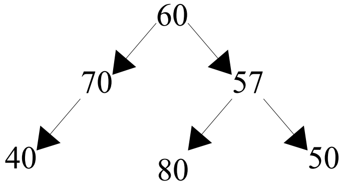


6. For each of the following trees, say whether it is a valid AVL tree and if not, why not.

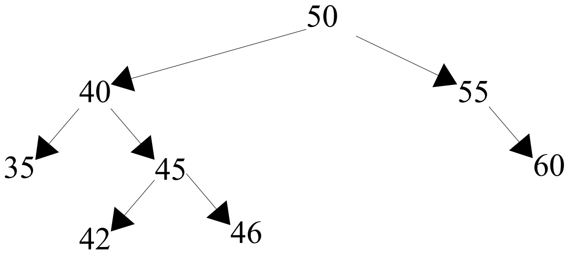
   A.



   B. (Be extra careful on this one.)

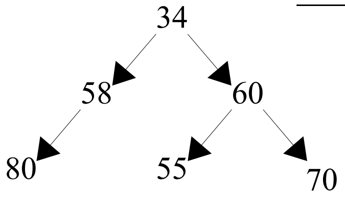


7. Mark the balance factors on the nodes of the following AVL tree. Then draw the tree that would result from inserting 47.



8.

A) Suppose you did an inorder traversal of the following tree. Note that it is not a Binary *Search* Tree. In what order would you visit the nodes?



____

B) Suppose you did a level-order (i.e., breadth first) traversal. In what order would you visit the nodes? _______________________________________

9. Suppose you were using a binary tree to represent a general tree in the usual way, with a Node's left child representing its first (leftmost) child and a Node's right child representing that Node's next sibling (the next child of the Node's parent). The following method takes a Node as it argument and returns true if the Node represents a leaf of the general tree. You may assume the argument, node, is not null. Finish the method.

```
public class Node<T>{
   Node<T> leftChild, rightChild;
   T data;
   public static boolean representsLeaf(Node node){ //assume node != null


     return __________________________________;
```