 name _____

CS 112 – Data Structures

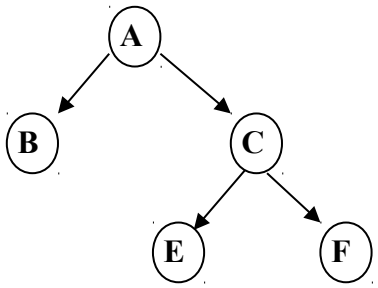
Summer, 2016

Midterm Practice Exam 2 - Answers

NOTE: See the topic list on Sakai > Resources > Exam Info > Midterm Exam 2 for a complete list of topics that may be on the actual exam.

- **Do not open** this exam until everyone has an exam and the instructor tells you to begin.
- 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 calculator watch).
- Write clearly – if we can't read or can't find your answer your, your answer is wrong.
- Make clear what is your answer versus intermediate work.
- **Please do not sit next to anyone whose name you know**

1. For the tree to the right



   a. What is its root? __ A ___ Its leaves ? __ B, E, F _

   b. What nodes are in the subtree rooted at C? _C, E, F__

   c. 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?

      No because C < E but node with E is left subtree of node with C

2. Finish the following recursive method. Its argument is the root node of a binary tree and it should return the sum of the data elements in all of the nodes of that tree.

```
public class BinTreeNode{
   public BinTreeNode leftSubtree, rightSubtree;
   int data;
   public static int sumData(BinTreeNode node){
       if ( node== null){

            return  0   ;
      }else{

            return node.data + sumData(node.leftSubtree)
                             + sumData(node.rightSubtree) ;
         }}
```

3. For the following class BinSchTreeNode, finish the method recBiggest so that it returns the largest data in tree. It must work recursively and efficiently. You may assume tree in not null and that it is a valid binary search tree.

```
public class BinSchTreeNode
  BinSchTreeNodeNode leftSubtree, rightSubtree;
  int data;
  public static int recBiggest(BinSchTreeNode tree){

   if (  tree.rightSubtree == null  ){

     return tree.data;

   } else {

     return  recBiggest(tree.rightSubtree);
}}
```

4. Finish the method itBiggest below. It is just like recBiggest above but it must be iterative and not recursive.

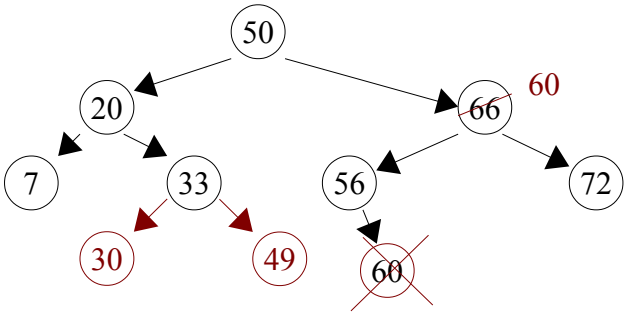
```
public int itBiggest(BinSchTreeNode tree){

BinSchTreeNode place = tree;

 while(  place.rightSubtree != null ){

    place = place.rightSubtree;  }
return place.data  ;}
```

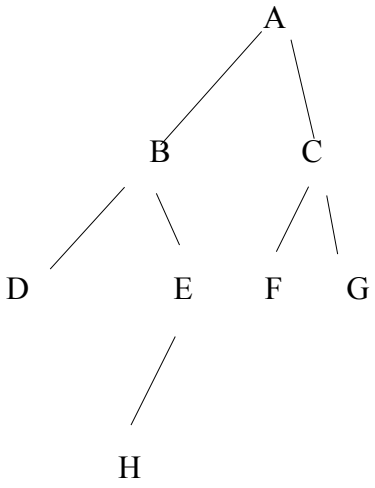
5. Suppose we do the following operations on the Binary Search Tree below. Mark the resulting changes on the tree:
insert 30
insert 49
delete 66



6.

a. For this tree, if we process the nodes in an in-order traversal, in what order will the nodes be processed?

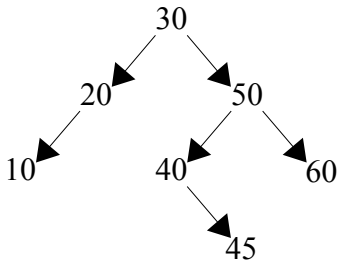
___D B H E A F C G_____



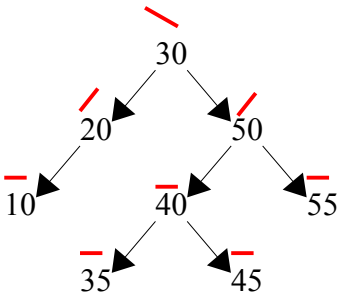
b. For the same tree. if we process the nodes in breadth first (level order) manner, in what order will the nodes be processed?

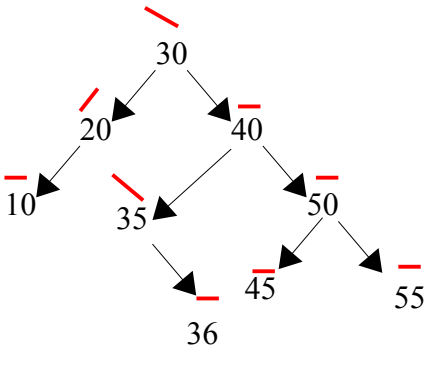
___A B C D E F G H_____

7. If you were searching the Binary Search Tree below for the target 41, which number or numbers would you compare with 41? 30, 50, 40, 45

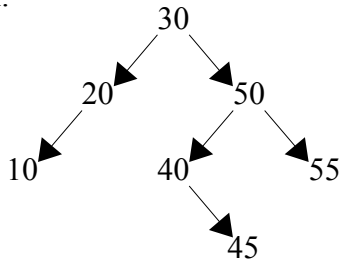


8. Mark balance factors in the nodes of the following AVL tree. Then insert 36, and draw the resulting AVL tree.

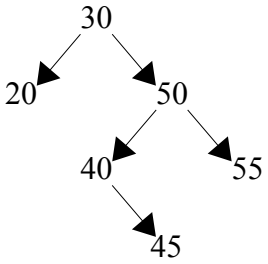




9. For each of the following trees, say whether it is a valid AVL tree and if not, why not.

A. 

valid

B. 

not valid, because the root node has a left subtree of height 0 and a right subtree of height 2, which differs by more than 1