CMSC 341 **Data Structures** **Spring 2013**
MW 4-5:15

- (1 point) Print your name on the cover of the exam booklet

1. (5 Points) Put the following growth functions in ascending order by their asymptotic growth rates:

   $n,\ n^{1.001}, \log n^2,\ n^2, \log^2 n, 100000,\ \sqrt{n}$.

2. (15 points) What is the asymptotic worst time performance (Big Oh) for each of the following code segments? Please draw a box around your final answer.

```
a) int sum=0;
   for (int i =1; i <= n; i++)
        for (int j = 0; j < 100; j++)
          sum++;

b) int sum=0;
   for (int i = 1; i <= n; i++)
        for (int j = 0; j*j < n; j++)
          sum++;

c) int sum=0;
   for (int i = 1; i <= n; i*=2)
        for (int j = 0; j < i; j++)
          sum++;
```

3. (30 points) Lists can be used to represent sets. Let lists L1 and L2 represent two sets of n1 and n2 elements of the same type, respectively. Assume that n1 and n2 are both positive, that there are no duplicates within each set, and that L1 and L2 are not necessarily in sorted order. You are asked to design an algorithm for the set difference operation L1 - L2. In other words, your code should keep all nodes in L1which do NOT appear in L2, and remove every node in L1 that DOES appear in L2. Note that the list L1 will likely be changed after this operation.
   a) (5 points) What implementation (ArrayList or LinkedList) would you choose for the lists? Justify your choice.
   b) (10 points) Write pseudo code for this set difference operation in your chosen implementation. (Pseudo-code is ordinary English, written as a series of steps as though a computer were able to execute them.)
   c) (5 points) What is the time performance of your code? Your answer should be given in Big-Oh of n1 and n2.
   d) (10 points) If both L1 and L2 are sorted, can you modify your code so that this operation can be done more efficiently? If your answer is yes, give the modified pseudo code and its time performance. If your answer is no, explain why.

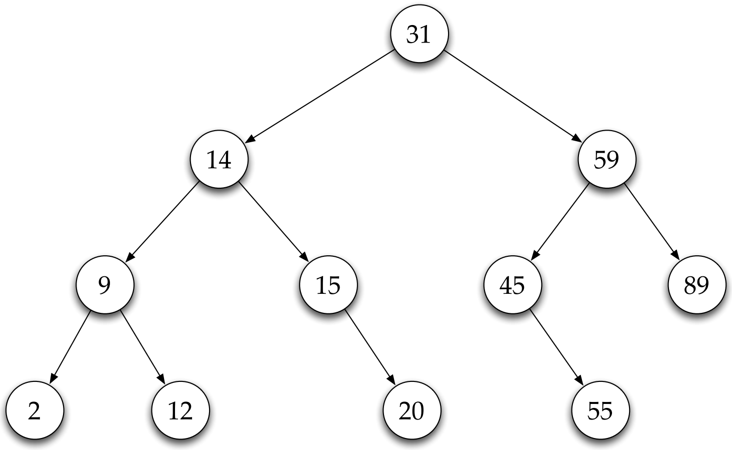
4. (10 points) The number of comparisons to build a BST of *n* elements is $O(n^2)$ in the worst case and $O(n \lg n)$ in the best case. Describe the best and worst case scenarios (what input would give these running times) and explain why this is so.

5. (10 points) Prove by induction that $\sum_{i=0}^{N} 2^i = 2^{N+1} - 1$ for all $N \geq 0$.

6. (5 points) Suppose the result of a post-order traversal of a BST is 10, 13, 12, 15, 25, 20, 14. Draw the original BST.
7. (10 points) Suppose we have a list L of *n* elements, stored using a singly-linked list. Write pseudocode that produces the reverse of L using a stack. Describe which stack implementation your algorithm would use (you do not need to give code for the implementation) and justify your choice. What is the complexity of your algorithm in terms of *n*, the number of elements in the original list?

8. (14 points) Consider the BST shown below

   a) (6 points) List the output from a *pre-order traversal*, an *in-order traversal*, a *post-order traversal*, and a *level-order traversal* of the tree.
   b) (4 points) Draw the new tree after deleting key 14 (and replacing it with it's predecessor) and then inserting key 39.
   c) (4 points) Draw the new tree after inserting key 14 back into the BST generated from b).


31
14
59
9
15
45
89
2
12
20
55