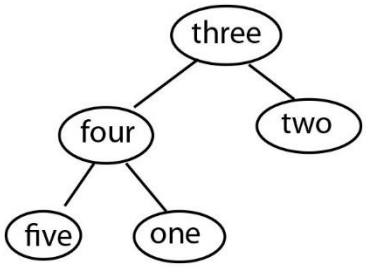
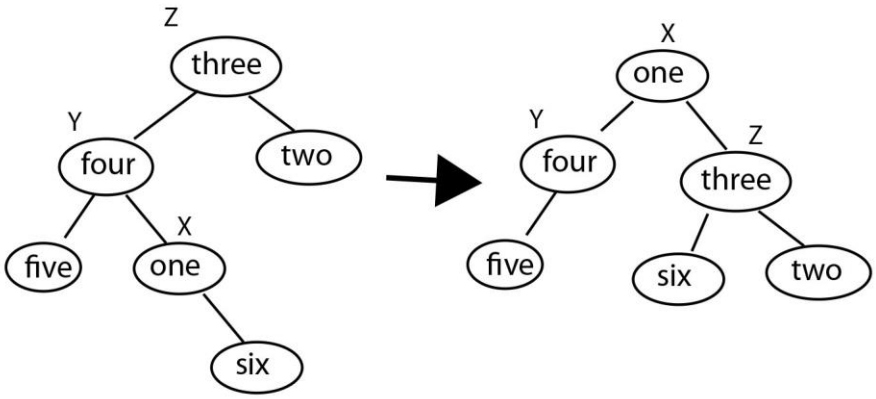
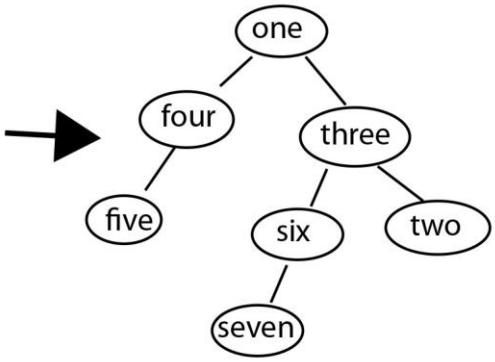
CSCI 151
Exam 2
April 2017

1. Here is an AVL tree . **Find the AVL tree we would get by adding the strings "six" and "seven" (in that order) to this tree**



Here is my solution:





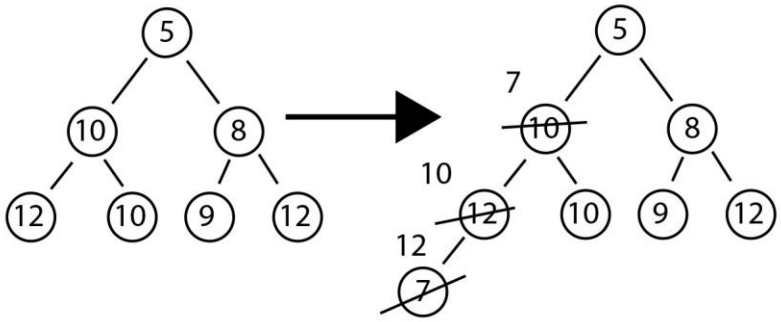
2. Here is a heap stored in an array:

	5	10	8	12	10	9	12			

a. What is the array that results from inserting the value 7 into this heap?

Answer:

	5	7	8	10	10	9	12	12		



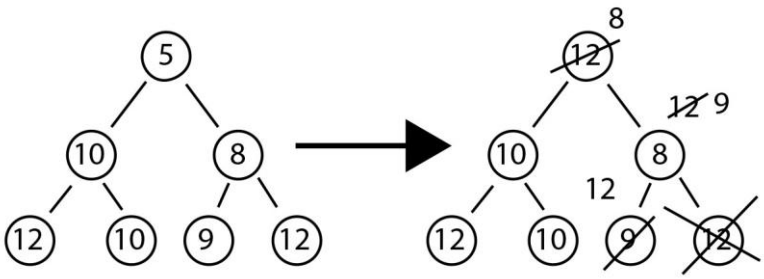
Now let's go back to our original heap:

	5	10	8	12	10	9	12			

b. What is the array that results from removing the smallest value in the heap?

Answer:

	8	10	9	12	10	12				



3.

a. Here are 5 data structures. **For each give a Big-Oh estimate of the running time for inserting a new element into a structure that already has n elements**:

Binary Search Tree: Average O(log(n)) Worst: O(n)

AVL Tree/TreeMap: Average = Worst = O(log(n))

Heap/Priority Queue: Average = Worst = O(log(n))

HashMap (Linked, as you implemented in Lab 8): Average = O(1) Worst = O(n)

Trie: O(1): Insertion is proportional to the size of the key; has nothing to do with n

b. Suppose you need to implement a mapping structure where the keys are strings, and one of the operations you need to write should return a list of the keys in alphabetical order. **Would this be easier for a HashMap structure or a TreeMap structure? Why?**

TreeMap. An inorder traversal gives the keys in alphabetical order.

4. Here is a class for Binary Search Trees that holds integer values

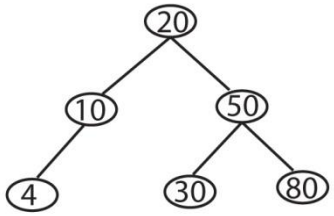
```
public class BST {
        Integer value;
        BST left, right;
        int size; // the number nodes in the tree with this as root
}
```

Below is a picture of a tree using this structure. **Write a method for this class**

Integer kthLargest (int k)

that returns the kth largest value, or null if k is greater than or equal to the size of the tree.

For the tree pictured kthLargest(0) is 4, kthLargest(1) is 10, kthLarges(2) is 20 and kthLargest(4) is 50.




```
Integer kthLargest( int k ) {
        int leftSize;
        if (left == null)
                leftSize == 0;
        else
                leftSize = left.size;
        if (leftSize == k)
                return value;l    else if (leftSize > k)
                return left.kthLargest(k);
        else if (size > k)
                return right.kthLargest(k-leftSize-1);
        else
                return null;
}
```

You could also do this by writing a function that returns a list with an inorder traversal of the tree, and returning the kth value in that list.

5. I have a very large file containing information about actors and movies they have acted in. The first few lines are

    Humphrey Bogart###Casablanca
    Judy Garland###The Wizard of Oz
    Daisy Ridley###Star Wars: The Force Awakens
    Humphrey Bogart###The Maltese Falcon

    Let's assume you know how to get the actor/movie information from each line. The lines come in an arbitrary order; the movies for any particular actor are not listed together. There is also a problem with the data: because of the way the datafile was generated some lines appear more than once, and not together. The line: "Humphrey Bogart###Casablanca" might appear 5 times at difference places in the data file. You have been asked to write a program that reads this datafile and prints the names in order of the 50 actors who have acted in the most movies. Explain in English how you will do this. **Which of the standard Java data structures that we have discussed this semester (ArrayList, LinkedList, Stack, Queue, TreeMap, TreeSet, HashMap, HashSet, PriorityQueue) will you use and how will you use it?** You don't need to write any code for this question; just tell me in English how to put data structures together to solve this problem.

    There is no one structure that makes a good solution for this, so do it in two steps. First read the data file into a HashMap where the keys are the actors' names and the value associated with each actor is a set (either TreeSet or HashSet) of that actors movies. With a set you don't have to check for duplicates; that is done automatically. The second step, after all of the data is read, is to put each actor and the size of his/her set of movies into a PriorityQueue where the comparator compares the number of movies. Poll this queue 50 times to get the top 50 actors.