CS 151 Name _____
Final Exam
December 17, 2014

Note that there are 10 equally-weighted qeustions.

1. [20 points] Here is a list of data: 4 2 18 1 3 7 9 0 5. For each of the following structures I will walk through the list in order, add each item to the structure and then go into a loop in which I remove elements one at a time from the structure and print them as I remove them. In what order do I print the items for

   A) A stack

   5 0 9 7 3 1 18 2 4

   B) A queue

   4 2 18 1 3 7 9 0 5

   C) A priority queue

   0 1 2 3 4 5 7 9 18

   D) In the add stage, I form a graph in which the values are nodes, and there is an edge from any value to any value except itself that it divides evenly into (e.g., there is an edge from 2 to 4 and an edge from everything to 0). In the removal stage I do a topological sort of this graph.

   You need to list the nodes in an order where if a divides evenly into b then a comes before b in the ordering. Here is one such order:

   1 2 3 4 5 7 9 18 0

   E) In the add stage I form a hash table of size 9 (the data just fits; you don't need to resize the table) with linear open addressing, using each data value as its own hash code. In the remove stage I remove and print the data at index 0, then the data at index 1, then index 2, etc.

   18 1 2 3 4 9 0 7 5

2. a) Describe InsertionSort, MergeSort, and QuickSort in one or two English sentences each. Give the Big-Oh estimates of WORST case running times of each.

InsertionSort

InsertionSort keeps a sorted portion of the array or list and at each step inserts the next element into this sorted portion. This is $O(n^2)$.

MergeSort

MergeSort splts the list in half, recursively sorts each half, then merges the two sorted halves together. This is O(n*log(n))

QuickSort

QuickSort chooses one element to be the "pivot" and rearranges the list so that the elements less than the pivot are to its left and the elements greater than the pivot are to its right. It then recurses on the elements to the left of the pivot and recurses again on the elements to the right of the pivot. This is also O(n*log(n)).

b) Which of these would you tend to use for most sorting situations?

QuickSort

c) Is there any situation where you would use InsertionSort rather than the others? If so give an example.

There is no place where you need to use InsertionSort. For small amounts of data it doesn't really matter which you use.

d) Is there any situation where you would use MergeSort rather than the others? If so give an example.

No.

e) Is there any situation where you would use QuickSort rather than the others? If so give an example.

Always

3. Here is a method that looks through an array for duplicates and sometimes returns the correct answer:

```
boolean dupes( int [] A ) {
      Random rand = new Random();
      for (int i = 0; i < A.length; i++) {
            for (int j = 0; j < 10; j++{
                  int x = rand.nextInt(A.length);
                  if (x != i && A[x] == A[i])
                        return true;
            }
      }
      return false;
}
```

Let N be the length of array A. Give a Big-Oh estimate of the running time of dupes(A) in terms of N.

This is O(N). The outer loop runs N times; the inner one 10 times, so it is O(10*N) = O)(N).

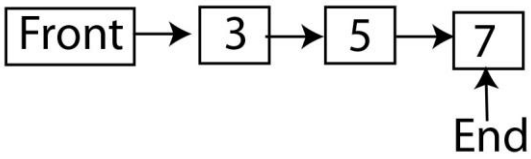
.

4. I have N integer values, where N is very large, stored in a file "Data.txt". Give Java code that opens and reads this file and prints the 100 largest values from the file, starting with the largest value. You can use any of the standard Java data structures for this. In case you don't recall the names of Java utilities for reading files, some of the are `Scanner, File, Scanner.nextInt(), Scanner.nextLine(), Integer.parseInt().`

```
class Comp implements Comparator<Integer) {
        int compare(Integer x, Integer y) {
                if (x > y)
                        return -1;
                else if (x < y)
                        return 1;
                else
                        return 0;
        }
}
void Number4() {
        PriorityQueue<Integer> P = new PriorityQueue<Integer>(100, new Comp() );
        Scanner scan = new Scanner (new File("Data.txt"));
        while (scan.hasNextInt() ) {
                int x = scan.nextInt();
                P.offer(x);
        }
        for (int  = 0; i < 100; i++) {
                System.out.println( P.poll );
        }
}
```

5. I want code for a Queue that holds Integer values using a linked structure.

a) Draw a picture of a Queue that holds the data [3, 5, 7].



b) Give the class header and data variables for your Queue structure

```
class Node {
        Integer data;
        Node next;
}
class Queue {
        Node Front;
        Node End;
}
```

c) Give the constructor for your Queue structure that initializes an empty queue.

```
public Queue() {
        Front = new Node();
        End = Front;
}
```

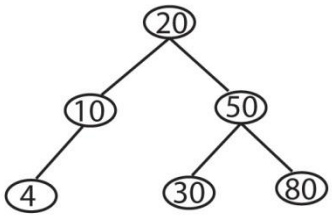
d) Give code for the method `void enqueue( Integer x)` that adds a new value to the queue.

```
 void enqueue( Integer x ) {
        Node p = new Node();
        p.data = x;
        End.next = p;
        End = p;
}
```

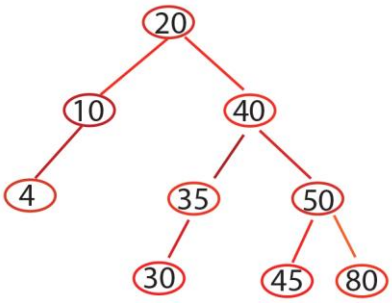
e) Give code for the method `Integer dequeue( )` that removes an element from the queue and throws a NoSuchElementException if you try to remove something from an empty queue.

```
Integer dequeue() throws NoSuchElementException {
        if (End == Front)
                throw new NoSuchElementException();
        else {
                Integer data =  Front.next.data;
                if (Front.next == End)
                        End = Front;
                Front.next = Front.next.next;
                return data'
        }
}
```

6. Here is a picture of an AVL tree. Give the tree that would result from adding the values 40, 35, and 45 (in that order) to this tree.



My solution:



7. I want to make a structure that will store notes to myself (these notes are just strings) by date. For example, the notes for a few days are

11/27/2014:	Eat turkey
	Watch football games
	Sleep a lot
12/17/2014:	Data Structures final
	Eat some ice cream to celebrate the Data Structures final being over
12/18/2014	Fly home

Define the classes and data structures that are needed to do this. For a class you just need to give the header and data variables. If you would use a standard data structure, say what classes it uses (such as Key or Value classes). Finally, give a sentence saying why you chose this particular data structure over the other options.

A straightforward way to do this would be a HashMap where the keys are integer arrays (holding the date) and the values are lists (either array lists or linked lists) of strings that hold the notes

8. I want to make an ArrayList of classes at Oberlin. Every class has a title, like “Data Structures”. Some classes have one instructor, like “Bob Geitz”. Some classes have two co-instructors, such as “Cynthia Taylor” and “Richard Salter”. What class would I use for the base-type of my ArrayList so that it can contain course titles and instructors of both single-instructor classes and two-insturctor classes? Give the class header and the class variables for the class or classes involved in this.

   There are many ways to do this. One way would be to have an abstract class Course and two concrete classes OneInstructorCourse and TwoInstructor course:

```
abstract class Course {
        String Title;
}

class OneInstructorCourse extends Course{
        String Instructor;
}

class TwoInstrucctorCourse extends Course {
        String Instructor1;
        String Instructor2;
}
```

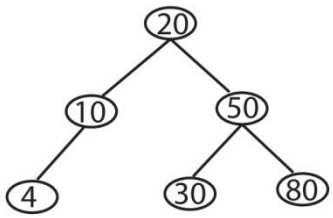
9. Here is a class for Binary Search Trees that holds integer values

```
public class BST {
        int 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 larger than the size of the tree. For the tree pictured kthLargest(0) is 4, kthLargest(1) is 10, kthLarges(2) is 20 and kthLargest(4) is50.




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

10. I have a large file of geneological data. Each line has the form

A | B | C

meaning that person C is the child of persons A and B. For example, one line might be

Bill Clinton | Hillary Clinton | Chelsea Clinton

Give an algorithm that you would use to implement a program that would allow you to enter two names and say if one is related to the other. You DON'T need to specify the kind of relation, just whether or not they are related in some way. Being children of siblings (i.e., cousins), or sharing a great-grandfather or being married to the sister of someone's grandmother are all examples of being related. For this question you can assume that every person has a unique name – there aren't two different people named "John Smith", for example.

One way to do this is to make an undirected graph of the people. For each line of the data file of form A | B | C have edges between A and B, B and C, and A and C. (Or you could use a directed graph; instead of an edge between A and B have an edge from A to B and another from B to A.) Then, to see if person X is related to person Y do a shortest path algorithm starting at X; if node Y ever comes out of this, then X and Y are related.