Don’t panic. Channel your energy in positive ways! Just take a minute to skim the test and read all the questions before attempting, and first try to answer those questions/parts about which you feel most confident. The questions are roughly arranged in order of increasing difficulty. Do not be alarmed if some of the questions are easy and some are hard. Do NOT spend too much time on one question. Pace yourself @ 100 points/75 minutes.

You may bring lecture notes (with non-code annotations), assignment/APT instructions/code, Java syntax summary, and one sheet (front and back) of printed notes, and any handwritten notes (but no other resources). You may refer to any program text supplied in lectures or assignments (but can’t bring any other code printed or handwritten). You may not use any computers, calculators, cell phones, minions, or other lifeforms. When writing code you do not need to worry about specifying the proper import statements. If you don’t know the exact syntax, do the best you can to get partial credit. The programming logic is more important than minor syntax. Assume that all libraries and packages we’ve discussed are imported in any code you write. You may write any helper methods you would like in solving the problems.

1. (5 points) I still haven’t found what I am looking for!
   Consider the binary tree below.
   a. Show the path that you will follow if you search for K on the figure on the left.
   b. Show the result of inserting M and N in the figure on the right.

2. (12 points) Equate this!
   Write the following equations as Java assignments. For example, \( x = (a^2 + 2ab + b^2) \) can be written as \( x = a*a + 2*a*b + b*b \). Not that you can write \( m_1 \) as \( m1 \), etc. and assume that these are all type double and that you have \( \text{Math.sqrt(number)}, \text{Math.pow(base, power)} \), etc. available. (2 points for each line)
   a. \( c = \sqrt{a^2 + b^2} \) \( \quad c = \text{Math.sqrt(a*a + b*b)}; \quad \) // or \( \text{Math.sqrt(Math.pow(a, 2) + Math.pow(b, 2))} \)
   b. \( w = \log xy + \log z \) \( \quad w = \text{Math.log(x*y) + Math.log(z)}; \quad \) // or \( \text{Math.log(x*y*z)} \)
   c. \( z = \frac{2x - 1}{x} \) \( w = (2*x-1)/x; \)

3. (8 points) Can you handle the truth?
   Fill out the following table with boolean values of true/false.

| a   | b   | !a  | a && b | !a || a | !b || b | !!(a || !b) && (a || b) | Perhaps others |
|-----|-----|-----|--------|--------|--------|------------------------|----------------|
| true| true| true| true   | true   | true   | true                   | true           |
| true| false| true| true   | true   | true   | true                   | true false     |
| false| true| true| true   | true   | true   | true                   | true false     |
| false| false| true| false  | true   | false  | false                  | false          |

4. (15 points) Big Oh time
   Give the running time of the following code segments as in terms of big-Oh if \( n \). Also, give values of \( m \), \( p \), and \( x \) when they are invoked with \( n = 50 \). Assume that initially \( m \), \( p \), and \( x \) are defined as follows before every code segment.

The questions in the actual exam will be different – so understand the concepts. Memorizing the answers to these questions will not work!
int m = 1000; int p = 100; double x = 64.0;

// Note: Even though the values are calculated with n = 50 (a constant), please express your running time as function of n, i.e. O(f(n)), considering that it can be set to any value.

a. (4 pts) Code segment 1
   for (int i = 1; i < n/20; i++) {
      x = x/2;
      m = m*2;
      p++;
   }
   Running time = O(n) m = 2000 p = 101 x = 32.0

b. (2 pts) Code segment 3
   for (int i = 0; i < 5*n; i++) {
      for (int j = 1; j < n; j=j*2) {
         m = p++;
      }
   }
   Running time = O(n log n)

c. Others like this...

d. (3 pts) In the code segments above, circle a, b, c, or d based on the rank order the running times of the code segments.
   i. Fastest a b c...
   ii. 2nd fastest a b c...
   iii. 3rd fastest a b c...

5. I love APTs
   Go through your APTs. Also check out http://www.cs.duke.edu/csed/newapt/johnsort.html.
   Understand primitive, arrays, ArrayList, if, for, while, getting max/min, defining/calling methods, static, Maps, HashMap, etc. If there are any string manipulation exercises, they would be very simple.
   Also understand the constraints that a method should check (such as resizing to ½ the array size, input being non-negative for factorial, etc.).

6. (20 points) Copy this!
   I want to take a linked list and copy it using Queues (see interface below).
   a. (10 pts) By focusing on the problem deduce the lower bound and give your reasoning behind your answer.
   b. (10 pts) Assume that you have Stack and Queue classes available with the following interfaces:

Please fill in the code on the next page to copy the linked list.

Note: There other ways to do this but you need to show you can use Stacks and/or Queues.

// assume Node class below
public class Node {
   String value;
   Node next;
   public Node(String first) {
      value = first;
      next = null;
   }
}
public class LinkedLister {
    public Node copyCat(Node inList) {
        // Declare local variables
        Node head;
        Queue<String> queue = new PriorityQueue<String>();

        // Add items to the queue
        while (inList != null) {
            queue.add(inList.value);
            inList = inList.next;
        }

        // Dequeue the items one at a time and add them to the list
        if (queue.isEmpty()) {
            return null;
        } else {
            head = new Node(queue.remove());
            Node current = head;
            while (!queue.isEmpty()) {
                Node newNode = new Node(queue.remove());
                current.next = newNode;
                newNode.next = null;
                current = newNode;
            }
        }
        return head;
    }
}