CS 2110 Placement Test

Students who wish to take the placement exam can pick it up during business hours no later than 3pm from the CS office in Rice Hall 527. They must complete and return the exam within 90 minutes. Students cannot use books, notes, computers, or help from other people while taking the exam. A student can only take the exam once, and students who have enrolled in CS 2110 at UVa for any amount of time are not allowed to take the placement exam. The exam will be graded in a few days, and the results will be emailed to the student. Students who pass do not receive credit, and will be required to take some other course at UVa in lieu of CS 2110.

Students should use the Java programming language on the test.

The placement test is made up of several multiple choice, short answer, true/false, implementation, and coding questions. Students interested in taking the test need to be familiar with:

- Basic Java Programing
    - Including knowledge of ArrayLists
- Java Classes
    - Fields/instance variables
    - Constructor (one or more in a class)
    - toString() method
    - equals() method
    - Getters/setters
    - Methods ("behavior")
    - Be able to declare, instantiate and initialize an object of a certain type (class)
    - Classes using other classes
- Basic Software Engineering
    - Phases of the software development lifecycle
    - Specifications and Requirements: focus on users, functional and non-functional requirements (and constraints)
    - Design: inheritance hierarchies, abstract classes, interfaces
- OO Design and Programming
    - Inheritance / understand the Object Class (in particular, the toString() and equals() methods…)
    - Interfaces
        - Comparable Interface // compareTo() method
        - Comparator Interface // compare() method
    - How to extend a class (inheriting methods, overriding methods, overriding abstract methods, use of super in constructor, use of super in other methods)
    - How Java decides which class's version of a method to invoke
- Sets and Maps
    - Set data structure, Map data structure, Collection interface
        - TreeSet / TreeMap / HashSet / HashMap
    - Java Collections Framework (general)
    - List, Set, Map interfaces
- Algorithms
    - Definition of an "algorithm" and how to tell if something is one
    - Asymptotic Analysis
        - Big-O
        - Comparison of common complexity classes (1, lg(n), n, n lg(n), n^2, n^3, 2^n)
        - Given code (non-recursive only), find its complexity class and identify its critical section
    - Searching and Sorting
        - Sequential search ("linear")

        - Binary Search: how it works. What's its complexity? (O(lg n)) How does this compare to sequential search? What must be true of the list before we use this? (Sorted)
        - Sorting: good sorts are O(n lg n). Be able to name the sorts in this complexity class (e.g. MergeSort)
- Data Structures
    - Trees definitions and common terms
        - Note recursive definition
        - Ability to draw trees and/or discuss illustrations of them
    - Tree traversal (in-order, pre-order, and post-order traversals on Binary Trees)
    - Heaps
        - Heap definition
        - Minheap/maxheap
        - How items are stored in a heap
        - Methods for heap: adding and removal (do not need to know how to code this)
- Grammars
    - Basic vocabulary and symbols
    - Interpret and describe components of a context-free grammar (CFG)
    - Constructing parse trees
    - Applying a sequence of derivation rules (production rules)
    - Understanding recursion in grammars