Computer Science 141 -- Exam 2

April 22, 2004

General Directions. This is an open-book, open-notes, open-computer test. However, you may not communicate with any person, except me, during the test. You have the full class period (75 minutes) in which to do the test. Put your answer to each question in the space provided (use the backs of pages if you need more space). Be sure to **show your work**! I give partial credit for incorrect answers if you show correct steps leading up to them; conversely, I do not give full credit even for correct answers if it is not clear that you understand where those answers come from. Good luck.

This test contains 5 questions on 5 pages.

Question 1 (15 Points). Imagine an "IntegerList" subclass of our "List" class. Instances of "IntegerList" are lists whose elements are integers. Show (pseudcode is OK) a recursive "countEven" method for "IntegerList;" "countEven" has no parameters, and returns the number of even integers in a list. For example, "countEven" sent to the list (2 4 5 8) would return 3.

Question 2 (15 Points). Imagine an “IntegerList” subclass of our “List” class. Instances of “IntegerList” are lists of integers. Here is a method for “IntegerList” that supposedly increments each element of a list by x:

```
// In class IntegerList:
public void increment( int x ) {
   if ( ! this.isEmpty() ) {
      int old = ((Integer)this.getAndRemove()).intValue();
      this.addItem( new Integer(old + x) );
      ((IntegerList)this.getRest()).increment( x );
   }
}
```

Prove that this algorithm really does replace every integer in the list with that integer's original value plus x.

Question 3 (15 Points). Here is an algorithm that prints the elements in a list from first to last, and then from last to first. For example, if run on the list (a b c), this algorithm would print

```
a
b
c
c
b
a
```

```
// In an "ExtendedList" subclass of List:
public void printTwice() {
   if ( ! this.isEmpty() ) {
      System.out.println( this.getFirst() );
      ((ExtendedList)this.getRest()).printTwice();
      System.out.println( this.getFirst() );
   }
}
```

Derive (and prove, if necessary), an expression for the number of primitive "List" messages (i.e., "isEmpty", "getFirst", and "getRest") that this algorithm sends, in terms of the length of the list.

Question 4 (20 Points). An “association list” is a list whose elements are themselves lists of two elements. For example, a simple association list might be written as ((a 1) (b 2) (c 3)). The two-element inner lists inside an association list “associate” their first element with their second. Thus, the preceding example associates a with 1, b with 2, and c with 3. One use for association lists is to find out whether one contains a particular association for an object – specifically, a “check(x,y)” message sent to an association list causes that list to look for an inner list whose first element equals x and whose second element equals y, and to return true if such a list is found, false otherwise. For example, the message “check('a',1)” sent to the example above would return true, while the message “check('a',3)” would return false.

Write (pseudocode is OK) a recursive “check” method for an “AssociationList” subclass of “List.” The “check” method takes objects x and y as its parameters, and returns true or false according to whether any inner list contains x as its first element and y as its second. You may assume as a precondition that the inner lists are all properly formed, i.e., they are lists of exactly 2 items.

Question 5 (10 Points). Phineas Phoole has decided to do our experiment that measures the lengths of lists of primes as a way of looking at how the number of primes less than or equal to n varies with n. Phineas has long believed that the number of primes less than or equal to n is proportional to the square root of n. He gathers the following data from his experiment:

n	***Number of Primes <= n***
9	4
25	9
49	15
81	22

Are these data consistent with Phineas's belief that the number of primes less than or equal to n is proportional to the square root of n? Why or why not?