# Computer Science 141 — Final Exam

December 15, 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 exam period (3 hours) in which to do the test, although it is designed for 2 hours. 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 7 questions on 8 pages.

**Question 1** (15 Points). A bank keeps track of customers' accounts in an ordered binary tree. The objects in this tree are “Account” objects, each of which represents one account, and contains, among other things, the amount of money in the account. Account objects return this amount in response to a “getAmount” message. Show (pseudocode is fine) a recursive algorithm that computes the total amount of money in all the bank's accounts. Write this algorithm as a method for a hypothetical “BankTree” subclass of our “OrderedTree” class.

**Question 2.** Here is an algorithm that does a strange computation:

```
static double strange( int n ) {
   // Precondition: n >= 0
   if ( n == 0 ) {
      return 1.0;
   }
   else {
      return 1.0/strange(n-1) + 2.0/strange(n-1);
   }
}
```

Part A (20 Points). Derive an expression for the number of division operations this algorithm executes, as a function of n. Be sure to show all your work, including proofs of closed forms.

(Part B on next page)

Question 2, Part B (10 Points). Prove that every invocation of “strange” returns either 1.0 or 3.0. Here is the algorithm again, for reference:

```
static double strange( int n ) {
   // Precondition: n >= 0
   if ( n == 0 ) {
      return 1.0;
   }
   else {
      return 1.0/strange(n-1) + 2.0/strange(n-1);
   }
}
```

**Question 3** (15 Points). Consider the following problem:

> Given a program, P, with no input, determine whether P will print "Hello, World" when run.

Show that no algorithm can solve this problem (i.e., that the problem is undecidable).

**Question 4** (15 Points). The following algorithm supposedly computes the product of all the numbers in a binary tree of numbers. Note that the product of no numbers is, by definition, 1.

```
// In a NumberTree subclass of OrderedTree...
public double product() {
   if ( this.isEmpty() ) {
      return 1.0;
   }
   else {
      return    ((Double)this.getRoot()).doubleValue()
              * ((NumberTree)this.getLeft()).product()
              * ((NumberTree)this.getRight()).product();
   }
}
```

Prove that this algorithm really does compute the product I claim it does.

**Question 5** (15 Points). Phineas Phoole suspects that it takes longer to insert the first item into a list than to insert the second. To test this hypothesis, he writes the following code:

```
List testList = new List();
long start1 = System.currentTimeMillis();
testList.addItem( “first” );
long end1 = System.currentTimeMillis();
long start2 = System.currentTimeMillis();
testList.addItem( “second” );
long end2 = System.currentTimeMillis();
System.out.println( “First insertion took ” + (end1-start1) + “ mS.” );
System.out.println( “Second insertion took ” + (end2-start2) + “ mS.” );
```

Whenever Phineas runs this program, it always reports times of 0 for both insertions.

Explain why you think Phineas's code reports times of 0, and show the code (pseudocode is OK) you would write to test his hypothesis.

**Question 6** (10 Points). Imagine that I am writing a program that plays some card game with its user. Two of the major classes I design for this program are named “ListOfCards,” and “Card.” “ListOfCards” represents a list of cards, and is a subclass of the “List” class you used in this course; I use it to represent a deck of cards, players' hands, etc. “Card” is a class that represents individual playing cards; each “ListOfCards” object will contain zero or more “Card” objects. Should I make “Card” a subclass of “ListOfCards?” Why or why not?

**Question 7** (20 Points). Below are some hypothetical execution times measured for different algorithms. Which, if any, of these algorithms do you think have execution times of $\Theta(2^n)$? Why?

| | Times (mS) | | |
|---|---|---|---|
| **N** | **Algorithm 1** | **Algorithm 2** | **Algorithm 3** |
| 1 | 6 | 20 | 0.8 |
| 2 | 10 | 80 | 1.2 |
| 4 | 50 | 320 | 2 |
| 6 | 200 | 720 | 8 |
| 8 | 760 | 1300 | 25 |
| 10 | 3050 | 2000 | 102 |

*Have a Nice Break!*