Computer Science 141 – Final Exam

May 16, 2003

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. 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 8 questions on 8 pages.

Question 1 (15 Points). Imagine a class `IntList` that is a subclass of our `List` class. Every item in an `IntList` is an integer. `IntList` should handle a parameterless message `filter`, which extracts from an `IntList` all those items that are greater than 17, and returns a new list containing only those items. For example, if the list [23 [4 [19 [17 []]]]] were sent a `filter` message, the result would be the list [23 [19 []]]. Write (pseudocode is fine) a recursive `filter` method.

Question 2 (20 Points). Prove that the following algorithm counts the number of leaves (nodes with no children) in a binary tree:

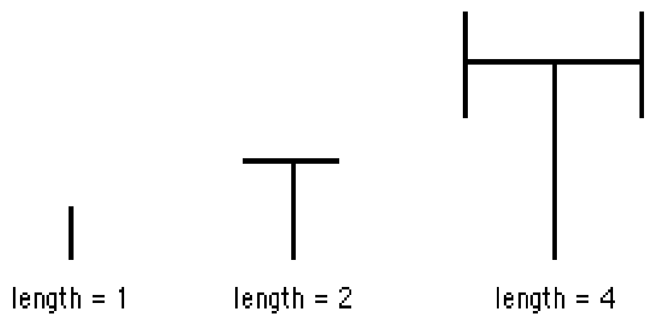
```
// In class ExtendedBinaryTree, which extends BinaryTree...
public int leaves() {
   if ( this.isEmpty() ) {
      return 0;
   }
   else if (    this.left().isEmpty()
             && this.right().isEmpty() ) {
      return 1;
   }
   else {
      return this.left().leaves() + this.right().leaves();
   }
}
```

Question 3 (15 Points). Let a “T-Figure” be a line drawing defined in terms of a length (which is a positive real number), as follows:

If the length is one or less, then the T-Figure is just a straight line of that length.

If the length is greater than one, then the T-Figure is a straight line of the specified length, with two T-Figures of half that length perpendicular to, and branching from, one end.

Here are some examples of T-Figures:



Give an algorithm (pseudocode is fine) that draws a T-Figure. Your algorithm should take the length of the figure as a parameter. You may phrase the algorithm as a `TFigure` method for a subclass of our `LineDrawing` class (or a similar class, you needn’t remember exactly how `LineDrawing` works).

Question 4 (10 Points). The following algorithm finds the largest item in a non-empty binary search tree:

```
// In class BinarySearchTree, which extends BinaryTree...
public Object largest() {
   if ( this.right().isEmpty() ) {
      return this.root();
   }
   else {
      return ((BinarySearchTree)this.right()).largest();
   }
}
```

What will be the asymptotic execution time of this algorithm in a typical binary search tree, in terms of the total number of items in that tree? Why?

Question 5 (15 Points). Imagine an `IntList` subclass of `List`. All items in an `IntList` are integers. The following algorithm determines whether any **subset** of the integers in a non-empty `IntList` adds up to *k*. The key ideas in this algorithm are that the items in a one-item list add up to *k* if and only if the one item equals *k*, and a subset of the items in a longer list add up to *k* if either a subset of the items in the tail do, or a subset of the items in the tail add up to *k* minus the first item (by adding the first item to that sum you get a total of *k*).

```
// In class IntList, which extends List...
public boolean sumsTo( int k ) {
   if ( this.rest().isEmpty() ) {
      // A 1-item list, so it adds up to k if its one and
      // only item is k, otherwise it doesn’t add up to k
      return ( this.first() == k );
   }
   else {
      // A longer list, a subset adds up to k if either a
      // subset of the tail does, or a subset of the tail
      // adds up to k – first.
      return ((IntList)this.rest()).sumsTo( k )
         || ((IntList)this.rest()).sumsTo( k–this.first() );
   }
}
```

Derive an expression for the number of arithmetic operations (the “==” and “-” operations) that this algorithm executes, in terms of the length of the list.

Question 6 (10 Points). Here are some measured running times for an algorithm that is supposed to execute in Theta(2^n) time, where *n* is the size of the algorithm's input.

Input Size (*n*)	**Time (seconds)**
1	8
2	36
4	160
6	384
8	640

Are these times consistent with the Theta(2^n) hypothesis? Why or why not?

Question 7. Suppose I am curious about how long it takes one of our `LineDrawing` objects to draw a line. So I write the following Java fragment:

```
LineDrawing tester = new LineDrawing();
long start = System.currentTimeMillis();
for ( int i = 0; i < 1000; i++ ) {
  tester.setPosition( 0.0, 0.0 );
  tester.movePen( 100.0 );
}
long end = System.currentTimeMillis();
System.out.println( “Main loop time = “ + (end-start) );
long cStart = System.currentTimeMillis();
for ( int i = 0; i < 1000; i++ ) { }
long cEnd = System.currentTimeMillis();
System.out.println( “Empty loop time = “ + (cEnd–cStart ) );
```

Part A (15 Points). There is a systematic error source in this code, in that the code will not measure the time to draw a line as accurately as it could. What is the error source? Show how you would modify the program to remove or compensate for it.

Part B (10 Points). Suppose you fix the error alluded to in Part A, and run the program. It prints

```
Main loop time = 501
Empty loop time = 1
```

What do you estimate the time to draw one line to be?

Question 8 (10 Points). One can sort using binary search trees in a straightforward way: start with an empty binary search tree, insert each item you want sorted, and then output the contents of the tree in left-to-right order. Suppose you are defining a `Sorter` class based on this idea. Further suppose that you have access to our `BinaryTree` class, or something equivalent. You can assume that `Sorter` objects have some way of accepting data to be sorted from clients, and some way of returning the sorted data sets to clients, but exactly what those mechanisms are doesn't matter for this question.

Would you define `Sorter` to be a subclass of `BinaryTree`, or would you define it as a class that uses `BinaryTree` member variable(s)? Why?