CMSC132
Fall 2005
Midterm #2

Grader Use Only:

#1		(12)
#2		(6)
#3		(24)
#4		(14)
#5		(7)
#6		(17)
#7		(8)
#8		(12)
Total		(100)
Honors		(14)

First Name: _____

Last Name: _____

Student ID: _____

Section time _____ TA: _____

I pledge on my honor that I have not given or received any unauthorized assistance on this examination.

Your signature: _____

General Rules (Read):

- This exam is closed book and closed notes.
- If you have a question, please raise your hand.
- Total point value is 100 points.
- Answer True/False questions by circling the T or F at the end of the question.
- Answer multiple-choice questions by circling the letter (e.g., a, b) at the front of each choice.
- Answer essay questions concisely using 1 or 2 sentences. Longer answers are not necessary and are discouraged.
- **WRITE NEATLY**. If we cannot understand your answer, we will not grade it (i.e., 0 credit).
- **Honors section questions only count for credit for students in the honors section**.

Problem 1 Networking (12 pts)

A. (4 pts) True or False
   a. An internet address is a 32-bit number T or F
   b. A computer can have multiple ports T or F
   c. Connections are easier to use than packets T or F
   d. Clients wait for servers to initiate communication T or F

B. (8 pts) Short essays
   a. What are IP, UDP and TCP? Hint, provide a single-word answer. (Note I am not asking what IP, UDP and TCP stand for, or what IP, UDP and TCP do.).

   b. To which layer of the OSI network model do UDP and TCP belong? Describe the function of this layer.

   c. Name a Java socket class that supports TCP.

   d. Name two pieces of information needed to set up a socket connection.

Problem 2 Files, Scanner, and Generics in Java (6 pts)

C. (4 pts) Files & Scanner

a. What is the motivation for viewing files as streams in Java?

b. Write code to show how the scanner may be used to read in a series of numbers (ints) from a text file.

```
class Scanner {
        Scanner ( … ) { … }
        boolean hasNext( ) { … }
        String next( ) { … }
        int nextInt( ) { … }
}

myFileReader( ) {
          try {
                BufferedReader f = new BufferedReader( new FileReader( filename ));

                // insert code to read in numbers from file









          } catch (IOException e) {
                System.out.println(e.getMessage());
          }
}
```

D. (2 pts) Generic types
   a. Classes in the Java Collection Framework support generic types in Java 1.5. Show how to modify the following code to declare a TreeSet class for Doubles using generics, then add & get a Double object from the TreeSet.

```
// previously
TreeSet mySet = new TreeSet( );
mySet.add( new Double(1.0) );
Double d = (Double) mySet.get(0);

// new code using generic types
```

Problem 3 Algorithmic Complexity (24 pts)

E. (6 pts) Algorithmic complexity
   a. List 2 reasons analyzing complexity is better than benchmarking

   b. Describe the difference between average case and expected case

   c. Give an example of a recurrence relation

F. (4 pts) Big-O notation and complexity

a. What happens to the running time of a $O(n^2)$ algorithm when the problem size is increased by a factor of 2?

G. (6 pts) Calculating Big-O functions

What is the asymptotic complexity of the function f( ) below (using big-O notation) when the complexity of g( ), and h( ) are as shown?

```
f(n) {
  g(n);
  h(n);
}
```

a. $h(n) = n^2-3n+4$, $g(n) = 5n-7$ f(n) = O( )

b. $h(n) = 2n + 4\sqrt{n}$, $g(n) = 100$ f(n) = O( )

c. $h(n) = n \log(n)$, $g(n) = 2n$ f(n) = O( )

H. (8 pts) Finding critical sections

a. Provide a definition for critical sections.

Calculate the asymptotic complexity of the code snippets below (using big-O notation) with respect to the problem size n:

b.
```
for (i = 0; i < 5*n; i++) {
   for (j = 1; j < 50; j=j+2) {
            …
   }
}
```
f(n) = O( )

c.
```
for (i = 1; i < n; i=i*2) {
   for (j = 1; j < n; j=j*4) {
         ...
   }
}
```
f(n) = O( )

d.
```
for (i = 0; i < n-2; i++) {
   for (j = 0; j < n; j=j+2) {
         ...
   }
   for (j = 0; j < 2*n; j=j+1) {
         ...
   }
}
```
f(n) = O( )

Problem 4 Recursive Algorithms (14 pts)

I. (6 pts) Recursion

a. Provide a definition for recursive algorithms

b. What is the base case of a recursive algorithm?

c. Why is a base case needed in a recursive algorithm?

J. (8 pts) Recursive code

For each of the following codes, describe whether the function foo(n) will return a result when it is invoked. If no result is returned, explain why.

a.
```
int foo (int n) {
   if (n > 1)
      return 4+foo(n-1);
   return 1;
}
```
Returns result? T or F
If no result, why?

b.
```
int foo (int n) {
   return foo(n-2)
}
```
Returns result? T or F
If no result, why?

c. Write a recursive function to determine how many times a number k is found in an array of ints, returning the number of times k is found. You may write helper functions.

```
public static int findOccur(int[] array, int k)
```

Problem 5 Linear Data Structures (7 pts)

K. (3 pts) Taxonomy & properties

a. Elements in a linear data structure have exactly 1 successor T or F
b. Elements in a hierarchical data structure have exactly 1 predecessor T or F
c. We insert A, B, C into a stack. The first element to be removed is A. T or F

L. (4 pts) Given the following Java class definition for a singly linked list of ints, write code for the insert( ) method so that it inserts a node at the head of the list. Make sure that you maintain the correct value for "head" so it references the first node in the list.

```
Class Node {     int myValue;     Node next;          }

Class LinkList {
   Node head;                          // first node in list
   void insert (Node n) {
                                       // your code




   }
}
```

Problem 6 Trees (17 pts)

M. (6 pts) Binary search trees (BST)

a. What is a good reason for storing data in a binary search tree?

b. What is the average height of a binary search tree with n nodes when adding nodes in random order?

c. What is the average height of a binary search tree with n nodes when adding nodes in sorted order from largest to smallest?

N. (6 pts) Binary search trees examples

a. Draw the binary search tree created when the following values are inserted in order: 5, 7, 9, 11, 2, 8

b. Given the previous BST, draw the tree created when the node 7 is removed

O. (5 pts) Given the following Java class definition for a binary tree, write code to print out its nodes during a recursive inorder traversal of the tree, starting at the root. Assume "left" and "right" have the value "null" if no subtree is present. You may use helper functions.

```
Class Node {
   int myValue;
   Node left;
   Node right;
}

Class Tree {
   Node root;                  // root node in tree
   void inorderPrint( ) {      // print nodes in tree starting from root









   }
}
```

Problem 7 Heaps (8 pts)

P. (2 pts) Properties
   a. List two advantages to using heaps instead of binary search trees to store data.

Q. (6 pts) Algorithms
   a. Draw the heap created (in tree representation) when the following values are inserted in order: 8, 4, 2, 7, 5, 3 (assume heap has smallest element at top)

   b. Given the following heap (in array representation), show the resulting after the smallest element is removed (in array representation)

      [2, 5, 8, 6, 9]

Problem 8 Maps & Hashing (12 pts)

R. (6 pts) Sets & maps

a. Why can arrays and maps be considered similar (say when compared to a queue)?

b. Given the following TreeMap API, show how to write code to construct a TreeMap storing the population count for each city. The TreeMap should allow city names (Strings) to be used to look up city populations (Integers):

```
public class TreeMap {
   TreeMap( ) { … }
   Object get( Object key ) { … }
   Object put( Object key, Object value ) { … }
   Object remove( Object key ) { … }
}

TreeMap myDB = new TreeMap( );

void addCity( String CityName, Integer PopulationCount ) {
        … // write code to add population for city to TreeMap









}

Integer findPopulation(  String CityName ) {
        … // write code to find population for City









}
```

S. (6 pts) Hashing

a. It is legal for hash functions to produce 2 values for the same input T or F

b. It is legal for hash functions to produce the same value for 2 different inputs T or F

c. Describe the worst case behavior for a hash table using chaining (bucket hashing)

d. In Java, why will modifying the equals( ) for a class possibly require changes to the hashCode( ) method? What property must be preserved?

Problem 9 Honors Section (14 pts)

Credit is given only for Honors Section students!

T. (14 pts) Potpourri

a. A computer can have multiple internet addresses T or F

b. Reliable network connections guarantee data is delivered T or F

c. How are domain names such as cs.umd.edu translated into internet addresses?

d. Explain why we should or shouldn't use algorithmic complexity to compare two algorithms if the problem size is guaranteed to be between 10 and 1000

e. Give an algorithm complexity that is less complex than O($\sqrt{n}$)

f. Describe the main difference between a linked list and a doubly linked list

g. List an advantage of using a hash table instead of a binary search tree.

h. List an advantage of using a binary search tree instead of a hash table.