Problem 1 Networking

A. Network architecture
   a. What is a protocol?
   b. Why are protocols useful for a network?
   c. What is the difference between an IP address and a URL?

B. Reliability
   a. What is the difference between reliable and unreliable network connections?
   b. Why do we not simply make all parts of the network reliable?

C. Client-server model
   a. Servers can only serve one client T or F
   b. A client contacts a server first T or F
   c. How do clients distinguish between multiple servers at the same IP address?
   d. List examples each of 3 server and 3 client programs.

D. Java support for networking
   a. What are sockets?
   b. What is the main difference between the Socket and ServerSocket classes?
   c. What is the main difference between the Socket and DatagramSocket classes?
   d. Why is there no DatagramServerSocket class?
   e. In Java, why does data from a network appear similar to data from a file?
   f. What are applets?
Problem 2 Algorithmic Complexity

A. Algorithmic complexity
   a. What is algorithmic complexity?
   b. List 2 reasons benchmarking is better than analyzing complexity
   c. What is the difference between best case, worst case, and average case?

B. Big-O notation and complexity
   a. What does the Big-O notation represent?
   b. Why are $O(2n+4)$ and $O(n)$ considered equivalent?
   c. Why are $O(n^2)$ and $O(n)$ not considered equivalent?
   d. What are some simple rules for calculating Big-O notation?
   e. Sort the following complexity categories from least to most complex
      i. Constant $O(1)$
      ii. Cubic $O(n^3)$
      iii. Exponential $O(2^n)$
      iv. Linear $O(n)$
      v. Logarithmic $O(\log(n))$
      vi. Quadratic $O(n^2)$
   f. How are the following complexity categories similar?
      $O(n^2)$, $O(n^4)$, $O(n^3)$, $O(n^{12})$, $O(n^{99})$

C. Calculating Big-O functions

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

\[
f(n) \{ \\
g(n); \\
h(n); \\
\}
\]

\[g. \ h(n) = n+1, \ g(n) = 2n \quad f(n) = O(\ )
\]

\[h. \ h(n) = 3n, \ g(n) = 4n^2 \quad f(n) = O(\ )
\]

\[i. \ h(n) = \log(n), \ g(n) = 5\sqrt{n} \quad f(n) = O(\ )
\]

\[j. \ h(n) = 8\sqrt{n}, \ g(n) = 2n \quad f(n) = O(\ )
\]
D. Finding critical regions

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

k. for (i = 1; i < n; i=i+2) {
    System.out.println("Hello");
}

f(n) = O( )

l. for (i = 1; i < n; i=i*2) {
    System.out.println("Hello");
}

f(n) = O( )

m. for (i = 0; i < n; i++) {
    for (j = 1; j < n; j=j+2) {
        System.out.println("Hello");
    }
}

f(n) = O( )

n. for (i = 0; i < n-2; i++) {
    for (j = 0; j < 100; j=j+2) {
        for (k = 1; k < 3*n; k++) {
            System.out.println("Hello");
        }
    }
    for (j = 0; j < n; j=j+1) {
        System.out.println("Hello");
    }
}

f(n) = O( )

o. for (i = 0; i < n; i=i*2) {
    for (j = 1; j < n; j++) {
        System.out.println("Hello");
    }
}

f(n) = O( )

p. for (i = 0; i < n-2; i++) {
    for (j = 0; j < n; j=j*2) {
        for (k = 1; k < 5000; k=k*5) {
            System.out.println("Hello");
        }
    }
    for (j = 0; j < n; j=j+1) {
        System.out.println("Hello");
    }
}

f(n) = O( )
Problem 3 Recursive Algorithms

A. Recursion
   a. Describe the difference between an iterative and recursive algorithm?
   b. What are the 2 main parts of a recursive algorithm?
   c. Name 4 requirements for recursive algorithms to work correctly.
   d. Name 2 advantages & 2 disadvantages of recursive algorithms

B. Legality of recursive code

   For each of the following codes, describe what result is returned when foo(n) is invoked. If no result is returned, explain why.

   a. int foo (int n) {
      return foo(n-1);
   }

   b. int foo (int n) {
      if (n == 1) return 1;
      return foo(n-1);
   }

   c. int foo (int n) {
      if (n == 1) return 1;
      return 1+foo(n-1);
   }

   d. int foo (int n) {
      if (n == 1) return 1;
      return foo(n);
   }

   e. int foo (int n) {
      if (n == 1) return 1;
      return foo(n-1)+f(n-1);
   }

C. Writing recursive code (you may use helper functions)

   a. Write a recursive function to search an unsorted array for a number k
      public static boolean findNum(int k, int[] array)
   b. Write a recursive function to calculate the sum of an array of ints
      public static int sumArray(int[] array)
   c. Write a recursive function to determine whether an array of ints is sorted (ascending)
      public static boolean sortedArray(int[] array)
Problem 4 Linear Data Structures

A. Taxonomy & properties
   a. Describe the main difference between linear and hierarchical data structures
   b. Describe the main difference between a linked list and an array
   c. Describe the main difference between a queue and a stack
   d. Describe a circular linked list

B. Given the following Java class definition for a singly linked list

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

Class LinkList {
   Node head; // first node in list
   Node find(int k) { … } // find node with myValue = k
   void insertHead(Node n) { … } // insert node n at the head
   void insertTail(Node n) { … } // insert node n at the tail
   void removeTail() { … } // remove node at the tail
}
```

Write the following code:

a. find( k ) – Find a node with myValue = k in a linked list
b. insertHead( n ) – Insert node n at the head of a linked list
c. insertTail( n ) – Insert a node n at the tail of a linked list
d. removeTail() – Delete the node at the tail of a linked list
e. Which 2 methods in the LinkList class can be used to implement a queue?
f. Which 2 methods in the LinkList class can be used to implement a stack?
Problem 5 Maps & Hashing

A. Sets & maps
   a. What is the difference between a set and a map?
   b. What is the difference between a map and a hash table?
   c. How are maps useful?

B. Hashing
   a. What is a hash function?
   b. What is a desirable property of hash functions?
   c. What is a perfect hash function?
   d. What is a collision?
   e. What is the difference between open addressing and chaining (bucket hashing)?
   f. What happens when an open addressing hash table is close to full?
   g. What is the relationship between the hashCode() and equals() methods in Java?

Problem 6 Regular Expressions

A. What is the language associated with the following Java regular expressions?
   a. \[^A-F\]
   b. House | Car
   c. \w+@[\w\.]*
   d. Apt[1-9]+

B. Write Java regular expressions for each of the following languages:
   a. Even number of a’s followed by odd number of b’s. Strings in the language are: aab, aabbb.
   b. The language defined by valid dates specified in the format MM/DD/YYYY