Problem 1. (10 points) Assume that you have a class called MyLL which is an implementation of a linked list of ints (using the class MyNode which has fields int val, MyNode next and methods getValue(), getNext()). Write a method for this class which determines whether the list is sorted from smallest to largest.

Answer 1.

```java
public boolean inOrder() {
    if (head == null)
        return true;
    for (MyNode nd=head; nd.getNext() != null; nd=nd.getNext())
        if (nd.getValue() > nd.getNext().getValue())
            return false;
    return true;
}
```

Problem 2. (10 points) Sort the following array using bucket sort. Show the array after each pass.

```
10101 01010 00101 11100 11000 11001 10111 10011
```

Answer 2.

```
01010 00101 10101 11100 11000 11001 10111 10011
00101 01010 10101 10111 10011 11100 11000 11001
00101 01010 10011 10101 10111 11000 11001 11100
00101 01010 10111 10101 10111 11000 11001 11100
00101 01010 10011 10101 10111 11000 11001 11100
00101 01010 10011 10101 10111 11000 11001 11100
```

Problem 3. (10 points) You have the found the following method in a class MyLL. What would be accomplished if you invoked this method with head as the parameter (ie theList.hmmm(head)). Do not just translate the method line by line, instead explain its purpose. Show the result of calling this method on a small linked list.

```java
public int hmmm(MyNode nd) {
    int c;
    if (nd == null || nd.getNext() == null)
        c = 0;
    else if (nd.getValue() <= nd.getNext().getValue())
        c = hmmm(nd.getNext());
    else
        c = 1+hmmm(nd.getNext());
    return c;
}
```
Answer 3. Reading the linked list from left to right, this method counts the number of times consecutive values in the sequence decrease. If this method is called on the list
head → 2 → 5 → 8 → 3 → 7 → 4 → 13 → 15 → 17 → 16 → null
then the value 3 will be returned (since 8 drops to 3, 7 drops to 4, and 17 drops to 16).

Problem 4. (10 points) Assume that you have an empty hash table with capacity 11, hash function h(k) = k % 11, and collision resolution scheme double hashing with d(k) = 1+(k^2 % 10). Perform the following operations on the table and draw the state of the table after each delete operation: insert(18), insert(27), insert(35), insert(15), delete(27), insert(4), insert(7), insert(2), delete(15), insert(16), insert(3), insert(5), delete(18).

Answer 4.

<p>| | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>2</td>
<td>35</td>
<td>*</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>35</td>
<td>*</td>
<td>16</td>
</tr>
</tbody>
</table>

Problem 5. (10 points) Alberto and Doris are both creating a data structure to contain information about owners of bicycles. Alberto intends to implement a hash table with separate chaining (each position in the table is a linked list). Doris intends to implement a vector using java.util.Vector. Under what circumstances should you prefer Alberto’s plan? Under what circumstances should you prefer Doris’s plan? What information would Alberto and Doris want to know before coding? Explain.

Answer 5. Alberto and Doris would want information about how the data structure would be used. For example: what operations would be frequent/infrequent, how many items would be held in the structure, and what sorts of patterns will the data have. These last two items are particularly important for Alberto.

Time: Alberto has typical performance O(1) (if the size of the table is chosen appropriately) and worst case performance O(n) for insert, delete, and find. Doris has O(1) for insert (at the front) and O(n) for delete and find.

Space: Alberto has a linked list for each position. How much/little space this uses depends upon the size of the array. Doris will have approximately the correct amount of space used regardless.

Coding: Alberto’s coding will be more complicated and longer than Doris’s (most of her work is already done, since she is using the prebuilt class).

Alberto’s plan is better when deletes and finds occur commonly (much faster) especially if he has some idea of the number/pattern of data items. Doris’s plan is better if speed is unimportant or deletes/finds are rare or there is no idea of the number of items.
**Problem 6.** (10 points) Start with an empty priority queue (where higher values have higher priority). Perform the following operations and draw the state of the priority queue after every dequeue operation: enqueue(22), enqueue(17), enqueue(19), dequeue(), enqueue(50), front(), enqueue(8), front(), enqueue(42), dequeue(), enqueue(45), front(), enqueue(47), front(), dequeue().

**Answer 6.**
Front: 19, 17
Front: 42, 19, 17, 8
Front: 19, 17, 8
Front: 45, 19, 17, 8

**Problem 7.** (10 points) You have found the following method in a class called MyStack (which is an implementation of a stack of ints). What is accomplished if you call this method? Do not just translate the code line by line; describe what it does at a high level. Show the result of running this code on a small example.

```java
public boolean unknown(int n) {
    boolean ans;
    if (n < 0)
        ans = true;
    else if (isEmpty())
        ans = false;
    else
        ans = unknown(n-pop());
    return ans;
}
```

**Answer 7.** If possible, this method removes items from the top of the stack until the total of the removed items is greater than the value passed and returns true. If not possible, it removes all items from the stack and returns false. For example, if the stack contains 45, 67, 22, 78, 31 (where 45 is the top item) and unknown(122) is called then the stack becomes 78, 31 and true is returned, since 45+67+22 = 134 > 122, but 45+67=112 < 122.
Problem 8. (10 points) Given the 3 sets $A$, $B$, and $C$ depicted below, fill in each of the 9 items below.

Answer 8.

$A = \{a,b,c,d,g,h,i,j\}$
$B = \{c,d,e,f,i,j,k,l\}$
$C = \{g,h,i,j,k,l,m,n\}$
$A \cup C = \{a,b,c,d,g,h,i,j,m,n\}$
$B \cap C = \{i,j,k,l\}$
$A - B = \{a,b,g,h\}$
$(A - B) \cap C = \{g,h\}$
$|A| = 8$
$(B \cap C) - A = \{k,l\}$