Mid-Term Test — Section L5101 (Evening)

Duration: *50 minutes* (8:10pm–9:00pm)
Aids Allowed: NONE (not even calculators)

Student Number: |_|_|_|_|_|_|_|_|_|

Last Name: ______________________________

First Name: ______________________________

Tutorial Section: (circle one)	Daniel W.	William	Rosalia	Albert	Ruth	Daniel T.
	MP 102	MP 103	BF 315	BF 323	LM 155	LM 157

*Do **not** turn this page until you have received the signal to start.*
(In the meantime, please fill out the identification section above, and read the instructions below *carefully*.)

This term test consists of 3 questions on 5 pages (including this one). *When you receive the signal to start, please make sure that your copy of the test is complete.* Answer each question directly on the test paper, in the space provided, and *use the reverse side of the pages for rough work.* (If you need more space for one of your solutions, use the reverse side of the page and indicate **clearly** which part of your work should be marked.)

Be aware that concise, well thought-out answers will be rewarded over long rambling ones. Also, unreadable answers will be given zero (0) so write legibly. In your answers, you may use any facts given during the course, without proof or justification, *as long as you state them clearly*. You must justify any other facts needed for your solutions.

General Hint: We were careful to leave ample space on the test paper to answer each question, so if you find yourself using much more room than what is available, you're probably missing something. Also, remember that hints are just hints: you are not required to follow them!

1: ______/15

2: ______/25

3: ______/15

TOTAL: ______/55

Good Luck!

Question 1. [15 MARKS]

Part (a) [2 MARKS]

Write your student number **legibly** at the top of every page of this test, *except page 1*.

Part (b) [6 MARKS]

For each statement below, check the appropriate box.

TRUE	FALSE	
☐	☐	With Abstract Data Types, the important question is not "what" (*e.g.*, what operations are implemented), but "how" (*e.g.*, how the operations are implemented).
☐	☐	If I have a class `A` and I want to make its instance variables available to subclasses in another package, I can achieve this by giving "default" accessibility to those variables.
☐	☐	A Java `interface` can specify instance variables.

Part (c) [3 MARKS]

Give an example of how one would instantiate a `Vector` (from package `java.util`) *without* having an `import` statement.

Part (d) [4 MARKS]

Give an example of how one would indicate that a method can throw an `Exception`.

Question 2. [25 MARKS]

Consider the following Java `interface`.

```
// A SortedList stores Strings in non-decreasing sorted order.
public interface SortedList {

    // Inserts the given String at the correct position in this list.
    public void insert(String data);

    // Removes from this list every String greater than or equal to 'lower'
    // and less than or equal to 'upper'.
    public void deleteRange(String lower, String upper);

} // interface SortedList
```

Part (a) [7 MARKS]

Fill-in a representation invariant for class `ArraySortedList` given below.

```
abstract public class ArraySortedList implements SortedList {

    private String[] items = new String[100];
    private int numItems = 0;

    ////  WRITE YOUR REPRESENTATION INVARIANT HERE.  ////









    abstract public void insert(String data);

    abstract public void deleteRange(String lower, String upper);

} // class ArraySortedList
```

Part (b) [18 MARKS]

Implement the `delete` method for class `LinkedSortedList` given on the next page, and write a complete method specification for the method. Recall that class `String` implements the interface `Comparable`, so that two `String`s can be compared using the `.compareTo()` method. You do not have to write internal comments for your code, but you can get part marks even if your code is incorrect as long as you have comments that clearly show you have the right idea.

Question 2. (CONTINUED)

Part (b) (CONTINUED)

```
class Node {
    public String data;
    public Node link;
    public Node(String d) { data = d; }
} // class Node

public class LinkedSortedList implements SortedList {
    private Node head;

    public void insert(String data) { /* code omitted to save space */ }

    ////  WRITE YOUR METHOD SPECIFICATION HERE, THEN IMPLEMENT THE METHOD.  ////






    public void deleteRange(String lower, String upper) {






    } // void deleteRange(String,String)
} // class LinkedSortedList
```

Question 3. [15 MARKS]

Consider the following code (set in two columns to save space).

```
public class Easy {
    static int first(int x) {
        if ( x > 1 )             // line 1
            x = second(x - 2);   // line 2
        return x;                // line 3
    }

    static int second(int x) {
        if ( x > 1 )            // line 1
            x = third(x - 2);   // line 2
        return 2 * x;           // line 3
    }

    static int third(int x) {
        if ( x > 1 )            // line 1
            x = first(x - 2);   // line 2
        return 3 * x;           // line 3
    }

    public static void main(String[] args) {
        System.out.print(first(5) + " ");
        System.out.print(second(7) + " ");
        System.out.println(third(first(5)));
    }
} // class Easy
```

Part (a) [5 MARKS]

Write the output produced by running this code.

Part (b) [10 MARKS]

Draw the Call Stack at the point during the execution of the code just *before* line 3 of method `third` is executed for the **third** time.