University of Toronto
St. George Campus

Computer Science CSC 148S
Friday 2 March 2001

Mid-Term Test

Duration: *50 minutes* (10:10am–11:00am)
Aids Allowed: NONE (not even calculators)

Student Number:	____________________
Last Name:	____
First Name:	____
Lecture Section: (circle one)	François Pitt (MP 103) Allan Jepson (MP 203)
Tutorial Room: (circle one)	MP 203 SS 1086 BF 323 IN 204 UC 52 LM 123 LM 155 LM 157 SS 1070

*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 6 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 without justification any facts given during the course, *as long as you state them clearly*. You must justify any other facts needed for your solution.

General Hint: We were careful to leave ample space on the test paper to answer each question. If you find yourself using much more room than what is available, you're probably missing something, so you should stop and take the time to think about what you're doing.

# 1:	______/10
# 2:	______/10
# 3:	______/10
Bonus:	______/ 2
TOTAL:	______/30

Good Luck! PLEASE HAND IN

Bonus. [2 MARKS]

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

Question 1. [10 MARKS]

Complete the method `deleteSecondLast` according to the contract specified by **all** the comments in the LList class below.

```
    // Nodes for our linked list.
    public class LNode {
        int value;
        LNode next;
    }

    // Exception class for our linked list.
    public ListUnderflowException extends Exception { }

    // Linked list class
    public class LList {
        /* Representation invariant:  Either
         *    a) head = null and size =0,
         * or
         *    b) head != null and size = number of elements in
         *       the linked list.
         */
        private LNode head;
        private int size;

        // Delete the second last value in the linked list.
        // Throws ListUnderflowException if the list contains less
        // than two elements.
        public void deleteSecondLast() throws ListUnderflowException {
```

```
// ... continue deleteSecondLast on this page, if necessary.
```

```
        } // End of deleteSecondLast
    } // End of LList.
```

Question 2. [10 MARKS]

Consider running the main method in the following class.

```
public class DLNode {
    private static int num = 0;
    private int value;
    private DLNode next;
    private DLNode prev;

    public DLNode(DLNode prev, int value, DLNode next) {
        num++;
        this.value = value;
        this.prev = prev;
        this.next = next;
        prev = next;
        next = prev;
    }

    public static void moonwalk(DLNode n, int c) {
        if (c > 0)
           moonwalk(n.next, c-1);
        System.out.print(" "+n.value);
    }

    public static void main(String[] args) {
       DLNode root = new DLNode(null, 99, null);
       root.next = new DLNode(root, 66, null);
       root.next.next = new DLNode(root.next, 77, root);
       root.prev = root.next.next;

       // Line number 6.

       moonwalk(root, num);
       System.out.println();
    } // End of main
} // End of DLNode
```

[4] **(b)** What does the program print? (You may wish to do the next part of this question first.)

Question 2. (CONTINUED)

[10] **(c)** Sketch the memory model for the above program for the point the execution gets to "Line number 6" of the main method. To keep the sketch small, only draw the items for the given `DLNode` class. Include the run-time stack, the heap, and the static space in your sketch.

Question 3. [10 MARKS]

Consider the following `IntBTNode` class that can be used to store a binary tree of `int`s.

```
class IntBTNode {
   public int value;
   public IntBTNode left;
   public IntBTNode right;
   // Details of constructor intentionally omitted...
}
```

Write a *recursive* method `printAtDepth` that prints all the nodes of the tree that have a specified depth. Remember that the root has depth 1.

```
// Print each node of the tree which has depth d.  These nodes should
// be printed in left to right order.
// Precondition: d >=1, rt != null.
public static void printAtDepth(IntBTNode rt, int dep) {
```

```
}//end printAtDepth
```