Duration: **50 minutes**
Aids Allowed: **NONE**

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

**Last Name:** ____________________

**First Name:** ____________________

**Lecture Section:** __________ **Instructor:** ____________________

**Tutorial Room:** __________ **TA's Name:** ____________________

---

*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 midterm test consists of 3 questions on 5 pages (including this one), printed on one side of the paper. *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 the part of your work that 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 think that you need much more room that what is available, take the time to stop and think about what you are doing instead: there is probably a shorter, easier solution. Also, remember that hints are just hints: you are not required to follow them if you can think of a different solution.

Marking Guide

# 1: ______/12

# 2: ______/13

# 3: ______/15

TOTAL: ______/40

*Good Luck!*

## Question 1. [12 MARKS]

### Part (a) [2 MARKS]

Complete the "identification section" at the top of page 1, then write your student number **legibly** at the bottom of every page of this test except page 1 (where indicated).

### Part (b) [10 MARKS]

Consider the classes below that implement a "Ternary Search Tree". A Ternary Search Tree is a labelled tree with a branching factor of 3 that satisfies the following property: for every node `n` in the tree,

- every key in the left subtree of `n` is less than the key at `n`,
- every key in the middle subtree of `n` is equal to the key at `n`,
- every key in the right subtree of `n` is greater than the key at `n`.

(Obviously, duplicate keys are allowed in a Ternary Search Tree.)
Write the body of method `contains` in class `LinkedSimpleTST` below so that it meets its specification, *without using recursion*. Include appropriate internal comments.

```
class TSTNode {
    Comparable key;
    TSTNode left, middle, right;

    TSTNode(Comparable key) { this.key = key; }
}

public class LinkedSimpleTST {
    // The root of this TST.
    private TSTNode root;

    /* Assume that methods 'insert', 'delete', etc. are here... */

    // Return true if this tree contains key, false otherwise.
    public boolean contains(Comparable key) {




    } // end of contains(Comparable)
} // end of class LinkedSimpleTST
```

## Question 2. [13 MARKS]

Complete the method `deleteRange` below according to the contract specified by *all* the comments in the `LList` class. Include appropriate internal comments.

```
public class LNode {
    int value;
    LNode next;
}

public class ListUnderflowException extends Exception { }

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

    /* Assume that 'insert' and other methods are here... */

    // Delete n consecutive elements starting at position k of this linked list.
    // (Note: position 0 corresponds to the head of the list.)
    // Precondition: k >= 0, n >= 0, and the list contains at least k+n elements
    // Throws ListUnderflowException if the list contains less than k+n elements.
    public void deleteRange(int k, int n) throws ListUnderflowException {

    } // end of deleteRange(int,int)
} // end of class LList
```

## Question 3. [15 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) {
        // Point A.
        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;
        System.out.println(num);
        moonwalk(root, 3);
        System.out.println();
    }

} // end of class DLNode
```

### Part (a) [5 MARKS]

What does the program above print when it is compiled and run? This is not a trick question: the program does compile and run without error. (**Hint:** You may wish to do the next part of this question first.)

## Question 3. (CONTINUED)

**Part (b)** [10 MARKS]

Sketch the memory model for the program on the previous page at the point when the execution reaches "`// Point A.`" of the `moonwalk` method *for the first time*. To keep the sketch small, only draw the items for the given `DLNode` class in the object space. Include the run-time stack, the object space, and the static space in your sketch. (**Hint:** You may not need all of the space on this page.)

Total Marks = 40