**PROJECT 2. Linked Lists**

**ESE 344 Software Techniques for Engineers**

**SUNY at Stony Brook, Murali Subbarao , Weight: 14%**

**Revised DRAFT 3/25/2019 Subject to minor changes**

1. **Instead of Rectangle Class, if you make this project work for just integer data values, then you can get up to 12/14 points.**
2. **Instead of using iterators, if you make all data members of the list object to be public and use node pointers (instead of iterators) to pass as parameters to functions and complete part 2 of this project below, then you can get 13/14 points. (you can skip the implementation of iterator itr – and other methods that create compiler issues. In addition, if you implement this variation only for integer data values (instead of Rectangles) you can get upto 11/14 points for this project. You can pass pointers to nodes in the list to specify the position of items to be inserted and deleted. You implement ptr++ and prt—instead of iterator class.**
3. (8 points) Modify the doubly linked list implementation of LIST data structure in the text book to use singly linked lists where the items stored are Rectangles (modified version of Square class in Ch. 1). A few methods randomly chosen from your Class definition will be tested thoroughly by calling them from the main() function below. You should use Rectangle class, and iterators, to complete this part and the other parts. Your source code must be a modification of the source code in the text book.
4. Write a main() function that does the following. Your methods and functions should be general and should not assume the number/value of items.
5. (2 points) Read 27 pairs of double numbers (lij, wij) from input file that specify the length lij and width wij of 27 rectangles for list i and item j in list i, with i=1,2,3, j=1,2,3,…,9.

Create 3 lists and print them.

L1 : ((l11, w11) , (l12, w12) … ( l19, w19) [ has 9 rectangles ]

L2 : (l21, w21) , (l22, w22) … ( l29, w29) [ has 9 rectangles ]

L3 : (l31, w31) , (l32, w32) … ( l39, w39) [ has 9 rectangles ]

1. (2 points) Insert every element of L3 into L1 at the end if it is not already present in L1, and print L1.
2. (2 points) Delete every element of L3 from L2 if it is present in L2, and print L2.

Sample input file:

2.5 4.1 7.2 6.3 9.1 2.2 2.5 4.1 7.2 6.3 9.1 2.2 2.5 4.1 7.2 6.3 9.1 2.2

3.5 4.1 7.2 6.3 9.1 2.2 2.5 4.1 7.2 6.3 6.1 2.2 2.5 4.1 7.2 6.3 9.1 2.2

3.5 4.1 7.2 9.3 9.1 2.2 2.5 4.1 6.2 6.3 9.1 2.2 2.5 5.1 7.2 9.3 9.1 2.2

Sample output:

(b) L1:

2.5 4.1 7.2 6.3 9.1 2.2 2.5 4.1 7.2 6.3 9.1 2.2 2.5 4.1 7.2 6.3 9.1 2.2

3.5 4.1 7.2 9.3 6.2 6.3 2.5 5.1 7.2 9.3

(c) L2:

7.2 6.3 7.2 6.3 6.1 2.2 7.2 6.3