Lab Assignment #2 – Using Custom Data Structures, Generic Collections, LINQ, Lambdas and Delegates

Student: _____

Due Date: **Due Week 06** **Marks/Weightage: 30/10%**

Purpose: The purpose of this Lab assignment is to:

- Practice the use of Custom Data structures
- Practice the use of user defined Linked Lists, Stacks and Queues, Generic Collections, LINQ, Lambdas and Delegates.

References: Read the course's textbook chapter ppts, notes and class code examples (You can also refer previous chapters if you need to.) This material provides the necessary information you need to complete the exercises.

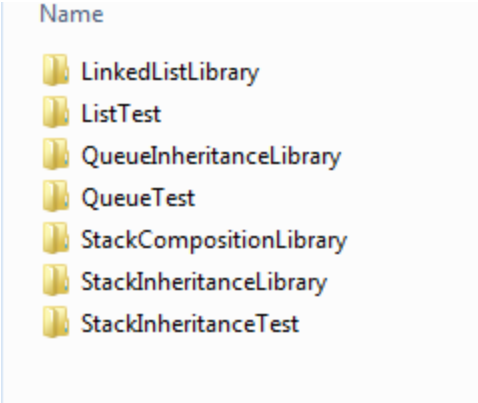
Instructions: Be sure to read the following general instructions carefully:

- This lab should be completed individually by all the students.
- The solution must be named using the first name and last name followed by Lab assignment number and section number. For the student name - John Smith, the solution name should be ***John-Smith_Lab02_Sec-001*** and project(s) name should be **John-Smith_Lab02_Ex1** for the first exercise, **John-Smith_Lab02_Ex2** for the second exercise, and so on**.**
- You will have to demonstrate your solution in a scheduled lab session and uploading the zipped solution/projects to the **Dropbox** link on **eCentennial**.
- You are required to follow the variable/control naming guidelines and must also implement exception handling in all the exercises.

Note (Very Important): *Late submission past due date is NOT allowed/accepted. You are required to be present during in-class demonstration.*

Exercise #1:

Refer the following solution folders /exercises (LinkedListLibrary) posted on e-centennial and which have been covered in the class.



Re-create a generic class (use T instead of object) LinkedListLibrary (.dll) by following the naming guidelines mentioned in the assignment and do the following enhancement/modifications:

a) You need to create a separate .cs file for each class.
b) Type of the data in the node should be double.(ListNode class)
c) Add following methods apart from the existing methods:
   1) **Search( )** method which searches a given element in the liked list. Use appropriate argument(s) for the method.
   2) **Count( )** method which count the number of elements in the linked list.
   3) **Avg()** method which returns the average of elements in the linked list
   4) Test it by creating and calling all the methods to demonstrate their working.

[10 marks]

Exercise # 1(a):

Re-create generic class (use T instead of object) **StackInheritanceLibrary** as per naming guidelines and modify it to maintain a stack of elements of type double. Also add one more method Peek() which returns the top element of the stack and test it by adding the reference of this library and demonstrate use of Push(), Pop() and Peek() methods. ***[5 marks]***

Exercise #1(b):

Recreate generic class (use T instead of object) **QueueInheritanceLibrary** as per naming guidelines and modify it to maintain a queue of elements of type string. (add few names) Test it by adding the reference of this library and calling methods for inserting and deleting names from the queue.
[5 marks]

Exercise #2: ***[10 marks]***

Querying an Array of Invoice Objects. Create an Invoice class which includes four properties – a PartNumber (type int), a PartDescription (type string), a Quantity of item being purchased (type int) and a Price(type decimal).
Perform the following queries on the array of Invoice objects and display the results:

a) Use LINQ to sort the Invoice objects by PartDescription
b) Use LINQ to sort the Invoice objects by Price
c) Use LINQ to select the PartDecription and Quantity and sort the results by Quantity.

Use the following sample data for Invoice class objects:

Part Number	Part Description	Quantity	Price
87	Electric Sander	7	57.98
24	Power Saw	18	99.99
7	Sledge Hammer	11	21.50
77	Hammer	76	11.99
39	Lawn Mower	3	79.50
68	Screw Driver	106	6.99
56	Jig saw	21	11.00