Student Name: **Weight:** **20%**

Student ID: **Marks:** **/100**

Assignment: Singly Linked Lists, Serialization and Testing

Background

A **linked list** is a linear data structure that is formed of a sequence of nodes. Each node contains two parts: Data and Address, which are dynamic and allocate memory as and when required. Insertion and deletion are easy to implement using linked lists, and items in a linked list can be accessed quickly. They are also used to implement other data structures such as Stack and Queue.

Serialization is the process of converting an object into a form that can be stored in a file, database or memory, or transferred across a network. Its main purpose is to save the state of the object so that it can be recreated when needed.

Unit Testing is a process of testing individual parts of a code to make sure that it works as expected. A unit test is a code implemented by a programmer that tests a small piece of functionality of a larger program.

Equipment and Materials

For this assignment, you will need:

- Visual Studio IDE

Instructions

1. Create the code for a program that meets the requirements described bellow.
2. Test your code against the expected output provided.
3. Submit the following to Brightspace:
    - GitHub URL for your program code
    - Zip file of the project folder.
    - Screenshots showing the results of the test cases.

      Note: Follow the folder structure outlined in Table 1.

Program Details

In this assignment, your goal is to implement an abstract data type (ADT) for a linked list.

- The SLL class must implement an ILinkedListADT interface.
- Each abstract method defined in the ILinkedListADT should be completely implemented, and any exceptions are to be appropriately propagated in the SLL class.
- The Node class represents a node in the linked list.
- Ensure that the linked list interface can perform the following functionality:
    - Prepend an item to the beginning of the linked list.
    - Append an item to the end of the linked list.
    - Remove an item at an index in the linked list.
    - Remove an item from the start of the linked list.
    - Remove an item from the end of the linked list.
    - Insert an item at a specific index in the linked list.
    - Replace an item in the linked list.
    - Get an item at an index in the linked list.
    - Get the index of an item in the linked list.
    - Check if the linked list has an item.
    - Clear all items in the linked list.
    - Get the number of items in the linked list.

    Important: Don't use already implemented classes, methods or libraries (such as LinkedList).
- Add at least one the following functionalities to the linked list:
    - Reverse the order of the nodes in the linked list.
    - Sort the nodes in the linked list into ascending order based on the names of the users.
    - Copy the values of the linked list nodes into an array.
    - Join two or more linked lists together to create a single linked list.
    - Divide the linked list into two separate linked lists based on a specific index position.

Place the interface (ILinkedListADT), the implementing class (SLL) and the Node class in the utility folder mentioned in Table 1.

Unit Testing

The **LinkedListTest** class is a test suite that performs unit testing on an implemented linked list. The following unit test cases will need to be implemented:

- The linked list is empty.
- An item is prepended.
- An item is appended.
- An item is inserted at index.
- An item is replaced.
- An item is deleted from beginning of list.
- An item is deleted from end of list.
- An item is deleted from middle of list.
- An existing item is found and retrieved.
- The additional functionality is working.

Important: Some of the test cases have already been implemented. Do not modify the implemented test cases. Develop and implement any **missing** test cases to achieve the needed code coverage.

Serialization

- The objects in the linked list SLL must be serialized, and the SLL object must be stored in binary format. You must be able to reconstruct the object from its binary form.
- Use the supplied **SerializationTests** class unit test cases to test for object serialization of the SLL.

  Note: Do not make any changes to the unit test suite.
- All the unit test cases need to pass with no errors. The objects are serialized to memory in the unit test suite and, therefore, no changes are made on the hard drive.
- Use the supplied User class in the **ProblemDomain** folder as the data type for the items in the linked list.
- Following the object-oriented principles, your project should contain ONLY the following classes and methods in their respective packages.

Table 1 – Package Folder Structure Outline

Folder	Class/Interface	Methods
Utility	ILinkedListADT	Add, AddLast, AddFirst, Replace, Count, GetValue, IndexOf, Contains, IsEmpty, Clear, Remove, RemoveFirst and RemoveLast
	SLL	Implementation of the above interface methods
	Node	Getters and setters
ProblemDomain	User	Getters, setters, and equals

Expected Output

All the test cases should pass.