**Lab Assignment – 4  
Total Points: 100**

**Due Date: Feb 12 (Thursday), 11.59 pm**

**Create a NetBeans project named lab04 and ensure it is saved to a location like desktop or your flash drive. In the project you will do the following:**

In this assignment you will be creating a generic version of ***Bag*** *(using a singly linked list)* which would be a collection of items that the client would define. The ***Bag*** interface will be a ***generic interface*** with a ***Type Parameter***. Just like in Lab 3, ***Bag*** will not order the items in any particular order and nor does it prevent any duplicates.

You will reuse the ***Bag*** Inteface from Lab03. Copy the interface and paste it in lab04 Project. Below is the specification of the Bag interface from lab03.

Develop a generic interface named ***Bag*** that can store certain number of items (type will be specified by the client). Provide the following methods in the interface.

* A method that returns the current count of items in the bag
* A method that checks if the bag is empty
* A method that checks if the bag is full
* A method that adds an item to the bag and returns true if the item was added successfully or false if the item was not added.
* A method that removes an item from the bag as long as the bag is not empty. This method would return the removed item from the bag.
* A method that removes a specific item from bag. This method will take as parameter the item to be removed, find the first occurrence of the item. Finally, the method returns true if the removal was successful else false.
* A method that removes all the items from the bag
* A method that returns the occurrences of a specific item in the bag.
* A method that checks if an item exists in the bag.

Design a generic class called ***LinkedBag*** which will implement the Generic ***Bag*** Interface created earlier. This class will include a ***Type Parameter*** as part of class declaration. The client of ***LinkedBag*** will be able to specify the actual type. The ***LinkedBag*** will use a ***singly linked list*** as the data structure to store the items. Inside the ***LinkedBag*** class do the following:

* First implement a ***private static Node class,*** inside ***LinkedBag*** class, whose object will form the nodes of the singly linked list. Each Node type of object should hold an ***element*** and a r***eference to the next Node***.
  + In order to allow a user to define the type of the element to be stored in the ***Node*** object, the ***Node*** class will require a generic type  
    ***private static Node <T>  
    {***

***……………………  
}***

* + Declare an instance variable of ***T*** type named ***element***
  + Declare an instance variable of ***Node<T>*** type called ***next***;
  + Implement a constructor that will take two ***parameters*** – ***a*** ***T type*** and ***Node<E> type*** to initialize the instance variables.
  + Provide ***get ( )*** and ***set( )*** methods for the instance variables.
* Declare an instance variable ***bagHead*** – of generic ***Node*** type in ***LinkedBag*** class.
* Declare an instance variable ***bagTail*** – of generic ***Node*** type in ***LinkedBag*** class.
* Declare another instance variable ***count***: This will provide the count of items currently stored in the bag. This count will increment as a new item is added to the bag and decrement as an item is removed from the bag.
* Provide a default constructor that will create an empty list.
* Implement the methods received from the ***Bag*** interface.
* The ***add ( )*** method should simply add an element to the beginning of the list.
* The ***remove ( )*** method remove and return the first node in the list. It will check whether the list is empty. If empty returns null else returns the reference to the removed first node of the list.
* The ***remove(T item)*** method removes a given item and returns true or false according to the success of the operation. This method will use the ***equals( )*** method to compare the contents of each of the node in the list with the contents of the parameter item. If there is a Node in the bag with the same contents then, it removes that item from the bag and returns true. The method returns false, If the bag is empty or, if the item is not in the bag. Use the following pseudo-code to develop the remove method.

*Locate a* ***Node N*** *that contains the item*

***if ( Node N exists)  
{***

***Replace the entry in Node N with the contents of the  
first Node in the list***

***Remove the first Node from the chain.***

***}***

* Implement the following additional methods
  + A method ***getItem(int i)*** that returns an item at a specific index position in the bag.
  + A method ***toArray( )*** that returns an array containing the copy of the items in the list.

Re-implement the ***Player*** class from lab – 3. You could simply copy and paste the contents of the Player class in Lab – 3 in a new class in this project folder.

Finally, create a Client Program ***NDSU-BasketBall*** with the ***main( )*** method. Inside the main method do the following:

* Create an object of ***ListBag*** called ***team*** to store all players’ information of NDSU Men/Women Basket Ball team.
* Run a ***for loop*** to prompt the user for each Player’s information, create the Player object and finally add the player to the team.
* Remove a player from the team.
* Add a new Player with some made up information.
* Display the current count of players in the team.
* Remove the Player that you just added earlier with made up information from the team using appropriate method.
* Display the current count of players in the team.
* Use a for loop to print the information of the Players in the team.
* Create an object of ***ListBag*** called courses to store the course ids of the courses that you are taking this semester (CSci 161, …..) as Strings.
* Populate the bag with the course ids.
* Remove a courseid from the Bag.
* Use a ***for loop*** to print the course ids from the bag.

Comment your ***Bag*** interface, ***ListBag*** and ***Player*** classes with Java Doc commenting style. Use single line or block style comments for the ***client*** program.

Draw a ***UML class diagram*** displaying all the classes and the relationships between them. The client class contents can be left blank with just ***main( )*** method displayed. This class will have association relationships between ***ListBag*** and ***Player*** class.

An association relationship simply means that a class uses another class – In this project, Client class is using ***ListBag*** and ***Player*** classes. This is depicted by an undirected broken line connecting the two classes.

**Things to turn in:**

* Open a Microsoft Word document name using the following file naming convention
  + i.e. *lab04-LnameFM*
    - lab04 = assignment prefix
    - Lname = your last name
    - F = your first initial
    - M = your second initial
* Copy and Paste the source code of the ***Bag*** Interface (make sure to use   
  *Ctrl + A* to select all the source code of the program and *Ctrl + C* to copy).
* Copy and Paste the source code of the ***ListBag and Player*** class.
* Copy and Paste the source code of the client program – ***NDSU-BaseketBall***
* Copy and paste the output of the client program
* Create a screen capture of your NetBeans IDE that includes the contents of the Output Window and paste it into your Word document below your source code.
  + To create a screen capture of your NetBeans IDE
    - Select, left-click in the NetBeansIDE
    - Use Alt-PrintScreen to place an screen capture image on the clipboard
    - Use Ctrl-V to paste the contents of the clipboard into your Word document
* Copy and paste the ***UML class diagram.***
* Next, zip the Project folder.
* Finally on blackboard, submit both your Word document and project zipped file.