CS 2003 Fundamentals of Algorithms and Computer Applications

Lab 3: Implementation of doubly linked list

Copy the following directory from your linux.ens account and `~class_sandip/2003/Lab03/*` which contains the following files:

- `ListException.java`
- `TestLR.java` (contains a main to test your work).
- `ListIndexOutOfBoundsException.java`
- `ListReferenceBased.java`
- `ListInterface.java`
- `Node.java`

This files contains code to implement a **single linked list** of **generic** types.

Modify the `ListReferenceBased.java` file to implement the `ListReferenceBased` class as a **doubly linked** list.

- You need to modify the `Node.java` file to convert the implementation for a singly linked list to that of a doubly linked list.
- You need to modify the `add` and the `remove` methods in `ListReferenceBased.java`.
- You must also implement the following missing methods in `ListReferenceBased` class as you did in the previous lab for the `ListArrayBased` class:

  contains returns either the index of the location in the list where the argument is present or -1 if the argument is not contained in the list,

  append appends the argument to the end of the list

  delete delete checks if a given item is contained in the list, and if it exists, deletes it.

  display displays the list items in sequence

  displayReverse displays the list items in sequence

  constructors add a constructor that creates a list with the one generic element as its only content and a copy constructor

- when you add a new method, write the corresponding javadoc comments.

You can create your own API specification using `javadoc`. Refer to the following website for additional information: `http://java.sun.com/j2se/javadoc/`. To generate the documentation, type `javadoc -d doc -private *.java` in the directory that contains your source files. This command will generate html files in a `doc` directory. You can use any web browser to see your documentation. For all the missing methods which do not have comments, write the javadoc comments.

Note the use of an inner class, `ListReferenceBasedIterator.java` (enables the list to be iterable, which allows the use of enhanced `for` loops.)

Submission: Submit through `WebCT` the files `Node.java` and `ListReferenceBased.java`.