Homework 6

Question 1 (10 pt.)

Implement a doubly linked list container by following the steps described below. Do your work on a directory named `hw6`. Once your code has been compiled and tested, remove all compiled "`.class`" files, create a package named `hw6.zip`, and upload it on Canvas.

a) (2 pt.) Create class `Node` in a file named `Node.java`. This class has the following properties:

- Public fields `previous` and `next`, pointing to the previous and next nodes in the list, respectively.
- Private field `data`, defined as a reference to an object of any type.
- Public constructor that takes the node's data as an argument.
- Public function `GetData()` that returns the node's data.

b) (6 pt.) Create class `DoublyLinkedList` in a file named `DoublyLinkedList.java`. This class has the following properties:

- Private fields `head` and `tail`, pointing to the first and last node in the list, respectively, or set to `null` if the list is empty.
- Private field `current`, set to the current node in the list, or to `null` to reflect a past-the-end position.
- Public functions `Head()` and `Tail()`, used to bring the current element to the first and last, respectively.
- Public function `Next()` used to move the current element one position forward. If the current element is the last, the new position will be past-the-end. If the current element is past-the-end, this call has no effect.
- Public function `Previous()`, used to move the current element one position backward. If the current element is past-the-end, the new position will be the tail. If the current element is the head, this call has no effect.

- Public function `Get()`, which returns the data associated with the current node, or `null` if the current position is past-the-end.

- Public function `Insert()`, which inserts a new element before the current element, or at the end of the list if the current position is past-the-end. This function takes one argument representing the data to be inserted in the list. After the insertion, the new element becomes the current element in the list.

- Public function `Print()`, which traverses the list and prints every element.

c) (2 pt.) Create class `Test` in a file named `Test.java`. This class contains a main program that performs the following actions:

- Instantiate a doubly linked list.

- Insert strings `“a”`, `“b”`, and `“c”` at the head of the list using three `Insert()` operations. The state of the list is now `[“c”, “b”, “a”]`.

- Set the current element to the second-to-last element with a call to `Tail()` followed by a call to `Previous()`. Then insert string `“d”`. The state of the list is now `[“c”, “d”, “b”, “a”]`.

- Set the current element to past-the-end with a call to `Tail()` followed by a call to `Next()`. Then insert string `“e”`. The state of the list is now `[“c”, “d”, “b”, “a”, “e”]`.

- Print the list with a call to `Print()` and verify that the state of the list is correct.