Collaboration policy: **Individual Assignment**
Total Points: **100**

Source Code

The following Java classes and/or interfaces are provided in one zip file attached to this Dropbox assignment:

- `DoublyLinkedList.java`
- `List.java`
- `Node.java`

In this assignment, **only** the `DoublyLinkedList` class will be modified by you. In particular, in the `DoublyLinkedList` class you **may only** modify the methods outlined in Part 1, and under no circumstances are you allowed to remove, add, or modify any other line of code in this class.

Furthermore, under no circumstances are you allowed to modify or create a new `List` interface or a new `Node` class. You must use these two files **as is**.

Lastly, you **may not** change the package structure! Specifically, `edu.cofc.csci230` cannot be removed or modified. If a solution is submitted with a different package structure, it will not be graded, no exceptions.

Part 1

In the `DoublyLinkedList` class please fully implement the methods listed below:

- `private void addNode(int index, Node<AnyType> t) throws IndexOutOfBoundsException`
- `private void setNode(int index, Node<AnyType> t) throws IndexOutOfBoundsException`
- `private Node<AnyType> removeNode( int index ) throws IndexOutOfBoundsException`
- `private Node<AnyType> getNode(int index) throws IndexOutOfBoundsException`
- `public void clear()`

In each method listed above there is a TODO comment - this is where you will add your code. Please note (and testing hint): the functionality of the above methods are identical to the ones in the List interface defined in the Java API.

Part 2

The `DoublyLinkedList` class provided to you has a main method. In this main please add test cases that demonstrate you have fully evaluated the operational correctness of the methods implemented in Part 1. To receive full credit, these test cases **must** be included.

Submission

Create a **zip** file that **only** includes the completed `DoublyLinkedList.java` file **Please do not include any folders (this includes the package folders) or any other project files in the zip file**. If you have any questions about the submission policy, you must resolve before the due date. Lastly, please plan appropriately, asking questions the day the assignment is due (within 12 hours) is too late. Please try to resolve any questions at least 2 days before the due date.

The name of the zip file must be your last name. For example, *ritchie.zip* would be correct if the original co-developer of UNIX (Dennis Ritchie) submitted the assignment. Submissions that use RAR compression algorithm (i.e. file that has the .rar extension), or any other compression algorithm other than zip, will not be accepted. Only assignments submitted in the correct format will be accepted (no exceptions).

Please submit the zip file (via OAKS) to the Dropbox setup for this assignment by the due date. You may resubmit the zip file as many times as you like, Dropbox will only keep the newest submission.

Per the syllabus, late assignments will not be accepted – no exceptions. Please do not email Luis or me your assignment after the due date, we will not accept it.

Grading Rubric

DoublyLinkedList Compiles	2.5 points
DoublyLinkedList Runs	2.5 points
Link traversal in the get() method	10 points
Thoroughness of your test cases	5 points
Instructor test cases (8 cases 10 points each)	80 points
	100 points

In particular, the assignment will be graded as follow, if the submitted solution

- Does not compile: 0 of 100 points
- Compiles but does not run: 2.5 of 100 points
- Compiles and runs with no errors: 5 of 100 points
- Thoroughness of your test cases: 10 of 100 points
- Passes all 8 test cases developed by the instructor: 90 of 100 points.
- Link traversals in get() method: 100 of 100 points