Lab #6: ArrayList and LinkedList

The main aim of the lab is to implement your own **ArrayList** and **LinkedList** with some basic methods similar to those in the Java Collection Framework.

Deadline: 23h59, 30/10/2023.

==========

Task 1: MyArrayList

==========

Task 1.1: Implements some basic methods in **MyArrayList.java** as follows:

```
public class MyArrayList<E> {
	public static final int DEFAULT_CAPACITY = 10;
	private E[] elements;
	private int size;

	public MyArrayList() {
		this.elements = (E[]) new Object[DEFAULT_CAPACITY];
	}

	public MyArrayList(int capacity) {
		this.elements = (E[]) new Object[capacity];
	}
// creates an array of double-size if the array of
elements is full
	public void growSize() {

	}
// Returns the number of elements in this list.
	public int size() {
		return 0;
	}

	// Returns whether the list is empty.
	public boolean isEmpty() {
		return false;
	}

// Returns (but does not remove) the element at index i.
	public E get(int i) throws IndexOutOfBoundsException
{
		return null;
```

```
    }

    // Replaces the element at index i with e, and
returns the replaced element. */
    public E set(int i, E e) throws
IndexOutOfBoundsException {
        return null;
    }

    // It is used to append the specified element at the
end of a list.
    public boolean add(E e) {
        return false;
    }

    // Inserts element e to be at index i, shifting all
subsequent elements later.

    public void add(int i, E e) throws
IndexOutOfBoundsException {

    }

    // Removes and returns the element at index i,
shifting subsequent elements earlier.
    public E remove(int i) throws
IndexOutOfBoundsException {
        return null;

    }
```

Then implements other methods as in Java Collection Framework - ArrayList.java

```
// It is used to clear all elements in the list.
    public void clear() {

    }

// It is used to return the index in this list of the
last occurrence of the specified element, or -1 if the
list does not contain this element.
    public int lastIndexOf(Object o) {
        return -1;
    }

// It is used to return an array containing all of the
elements in this list in the correct order.
```

```
    public E[] toArray() {
        return null;
    }

// It is used to return a shallow copy of an ArrayList.
    public MyArrayList<E> clone() {
        return null;
    }

// It returns true if the list contains the specified
element
    public boolean contains(E o) {
        return false;
    }

// It is used to return the index in this list of the
first occurrence of the specified element, or -1 if the
List does not contain this element.
    public int indexOf(E o) {
        return -1;
    }

// It is used to remove the first occurrence of the
specified element.
    public boolean remove(E e) {
        return false;
    }

// It is used to sort the elements of the list on the
basis of specified comparator.
    public void sort(Comparator<E> c) {

    }
```

==========

Task 2: MyLinkedList

==========

Task 2.1: Implements some basic methods in **MyLinkedList.java** (using a singly linked list) as follows:

Node.java:

```
public class Node<E> {// Generic
    private E data;
```

```
	private Node<E> next;

	public Node(E data) {
		this.data = data;
	}

	public Node(E data, Node<E> next) {
		this.data = data;
		this.next = next;

	}

//…

}
```

SinglyLinkedList.java:

```
public class SinglyLinkedList<E> {
	private Node<E> head = null;
	private Node<E> tail = null;
	private int size;

	public SinglyLinkedList() {
		super();
	}
// Returns the number of elements in the list.
	public int size() {
		return 0;
	}

	// Returns true if the list is empty, and false
otherwise.
	public boolean isEmpty() {
		return false;
	}

	// Returns (but does not remove) the first element in
the list.
	public E first() {
		return null;
	}

	// Returns (but does not remove) the last element in
the list.
	public E last() {
		return null;
```

```
    }

    // Adds a new element to the front of the list.
    public void addFirst(E e) {
    }

    // Adds a new element to the end of the list.
    public void addLast(E e) {
    }

    // Removes and returns the first element of the list.
    public E removeFirst() {
        return null;
    }

    // Removes and returns the last element of the list.
    public E removeLast() {
        return null;
    }
```