

ΕΡΓΑΣΤΗΡΙΟ 6

Δημιουργία-διαχείριση διπλά συνδεδεμένης λίστας

Άσκηση 6.1

Για να κατασκευάσετε μια διπλά συνδεδεμένη λίστα πρέπει να δημιουργήσετε τις παρακάτω κλάσεις:

- την **DoubleNode class**, η οποία περιγράφει έναν κόμβο μιας διπλά συνδεδεμένης λίστας (δίνεται στη συνέχεια) και
- την **DoubleLinkedList class** που υλοποιεί το **List interface**

Αρχείο DoubleNode.java

```
class DoubleNode
{
    private Object item;
    private DoubleNode next, previous;

    public DoubleNode() {
        this(null,null,null);
    }
    public DoubleNode(Object data, DoubleNode n, DoubleNode p) {
        item = data;
        next = n;
        previous = p;
    }

    public Object getItem() {
        return(item);
    }
    public DoubleNode getNext() {
        return(next);
    }
    public DoubleNode getPrevious() {
        return(previous);
    }
    public void setItem(Object newItem) {
        item = newItem;
    }
    public void setNext(DoubleNode newNext) {
        next = newNext;
    }
    public void setPrevious(DoubleNode newPrevious) {
        previous = newPrevious;
    }
}
```

```

public interface List
{
    public boolean isEmpty();

    public int size();

    public void insertFirst(Object data);

    public void insertLast(Object data);

    public void insert(int position, Object data) throws NoSuchListPosition;
    /* Τοποθετεί το νέο κόμβο στην υπ' αριθμό position θέση της λίστας. Αν το position
    είναι 0 ο κόμβος εισάγεται στην αρχή, αν το position είναι size() ο κόμβος εισάγεται
    στο τέλος, διαφορετικά αν position < 0 ή position > size() προκύπτει εξαίρεση */

    public Object removeFirst() throws ListEmptyException;

    public Object removeLast() throws ListEmptyException;

    public Object remove(int position) throws ListEmptyException, NoSuchListPosition;
    /* Διαγράφει τον κόμβο που βρίσκεται στην υπ' αριθμό position θέση της λίστας.
    Αν το position είναι 0 διαγράφεται ο πρώτος κόμβος, αν το position είναι size()-1
    διαγράφεται ο τελευταίος κόμβος, διαφορετικά αν position < 0 ή position >= size()
    προκύπτει εξαίρεση */
}

```

Να γραφεί πρόγραμμα (***DoubleListManagement.java***) για τη διαχείριση μίας διπλά συνδεδεμένης λίστας, η οποία περιέχει Strings. Η διαχείριση να γίνεται μέσω ενός μενού όπως το παρακάτω:

******* DOUBLE LINKED LIST MANAGEMENT *******

- 1- INSERT ELEMENT AT THE BEGINNING OF THE LIST
 - 2- INSERT ELEMENT AT THE END OF THE LIST
 - 3- INSERT ELEMENT AT POSITION **N** OF THE LIST
 - 4- DELETE ELEMENT FROM THE BEGINNING OF THE LIST
 - 5- DELETE ELEMENT FROM THE END OF THE LIST
 - 6- DELETE ELEMENT FROM POSITION **N** OF THE LIST
 - 7- LIST LENGTH
 - 8- IS THE LIST EMPTY
 - 9- PRINT LIST
 - 0- EXIT
- INPUT YOUR CHOICE (E.G. 5):

Άσκηση 6.2

Η **DoubleNode class**, η οποία περιγράφει έναν κόμβο μιας διπλά συνδεδεμένης λίστας μπορεί εναλλακτικά να αποτελεί μία κλάση επέκτασης (να επεκτείνει) της **SLListNode class** που ορίστηκε στα πλαίσια του εργαστηρίου 4:

Εναλλακτικό Αρχείο DoubleNode.java

```
public class DoubleNode extends SLListNode
{
    // Instance fields
    private DoubleNode previousNode;
    // Instance methods
    public DoubleNode getPreviousNode()
    public void setPreviousNode(DoubleNode n)
}
```

Συμπληρώστε την παραπάνω κλάση και δοκιμάστε το πρόγραμμα της άσκησης 6.1 με τον εναλλακτικό τρόπο.

Για επιπλέον εξάσκηση

Θεωρήστε ότι σας δίνεται μία λίστα ακεραίων αριθμών η οποία πρέπει να είναι ταξινομημένη:

Γράψτε κώδικα για τις δύο παρακάτω μεθόδους:

- **public boolean isSorted():** Ελέγχει αν η λίστα είναι ταξινομημένη σε αύξουσα τάξη
- **public void insert(Object data):** Δημιουργεί έναν καινούργιο κόμβο με περιεχόμενο data και τον τοποθετεί στην κατάλληλη θέση της λίστας ώστε αυτή να παραμείνει ταξινομημένη