Linked List Assignment

```
// This is the class representing a single node in a linked list
class Node {
    int data; // This is the data stored in the node
    Node next; // This points to the next node in the list

    public Node(int data) {
        this.data = data;
        this.next = null; // At the start, it doesn't point to any other node
    }
}

// This is the class for the linked list
class LinkedList {
    Node head; // This is the first node in the list (the head)

    // This method adds a new node at the beginning of the list
    public void addFirst(int data) {
        Node newNode = new Node(data);
        newNode.next = head; // The new node will point to the current head
        head = newNode; // The new node becomes the new head
    }

    // This method prints all nodes in the list
    public void printList() {
        Node current = head;
```

```
        while (current != null) { // Traverse the list till the end
            System.out.print(current.data + " ");
            current = current.next; // Move to the next node
        }
        System.out.println();
    }

    // This method deletes the first node in the list
    public void deleteFirst() {
        if (head != null) { // If the list is not empty
            head = head.next; // Move the head to the next node (removing the first one)
        }
    }
}

// This is the class representing a node in a doubly linked list
class DoublyNode {
    int data; // The data in the node
    DoublyNode next; // Points to the next node
    DoublyNode prev; // Points to the previous node

    public DoublyNode(int data) {
        this.data = data;
        this.next = null;
        this.prev = null;
    }
}
```

```
// This is the class for the doubly linked list
class DoublyLinkedList {
    DoublyNode head; // The first node in the list

    // This method adds a node at the beginning of the doubly linked list
    public void addFirst(int data) {
        DoublyNode newNode = new DoublyNode(data);
        if (head != null) {
            head.prev = newNode; // The current head's prev will point to the new node
        }
        newNode.next = head; // The new node's next points to the current head
        head = newNode; // The new node becomes the new head
    }

    // This method prints all nodes in the doubly linked list
    public void printList() {
        DoublyNode current = head;
        while (current != null) { // Traverse the list till the end
            System.out.print(current.data + " ");
            current = current.next; // Move to the next node
        }
        System.out.println();
    }
}

// Main method to run and test the linked list
```

```
public class Main {

    public static void main(String[] args) {

        // Testing the singly linked list

        LinkedList list = new LinkedList();


        // Add some nodes

        list.addFirst(10);

        list.addFirst(20);

        list.addFirst(30);


        // Print the list

        System.out.print("Singly Linked List after adding: ");

        list.printList();


        // Delete the first node

        list.deleteFirst();


        // Print the list after deleting

        System.out.print("Singly Linked List after deleting first node: ");

        list.printList();


        // Testing the doubly linked list

        DoublyLinkedList dList = new DoublyLinkedList();


        // Add some nodes

        dList.addFirst(40);

        dList.addFirst(50);
```

```
        dList.addFirst(60);



        // Print the doubly linked list

        System.out.print("Doubly Linked List after adding: ");

        dList.printList();

    }

}
```