Experiment-1.4

Student Name: Vishal Saini
UID: 23BCS10163
Branch: B.E-C.S.E
Section/Group: 23KRG-2A
Semester: 5th
Date of Performance: 19/08/2025
Subject Name: DAA
Subject Code: 23CSH-301

1. **Aim:** Apply the concept of Linked list and write code to Insert and Delete an element at the beginning and atend in Doubly and Circular Linked List.

2. **Objective:** To understand doubly and circular linked list

3. **Input/Apparatus Used:** Doubly and circular Linked List is used.

4. **Procedure/Algorithm: Pseudocode:**

Procedure for beginning of circular linked list:
Step1. Create the new node
Step2. Set the new node‟s next to itself (circular) Step3. If the list is empty,return new node.
Step4. Set our new node‟s next to the front. Step5. Set tail‟s next to our new node.
Step6. Return the end of the list.

Procedure for end of circular linked list:
Step1. Create the new node
Step2. Set the new node‟s next to itself (circular) Step3. If the list is empty,return new node.
Step4. Set our new node‟s next to the front. Step5. Set tail‟s next to our new node.
Step6. Return the end of the list.

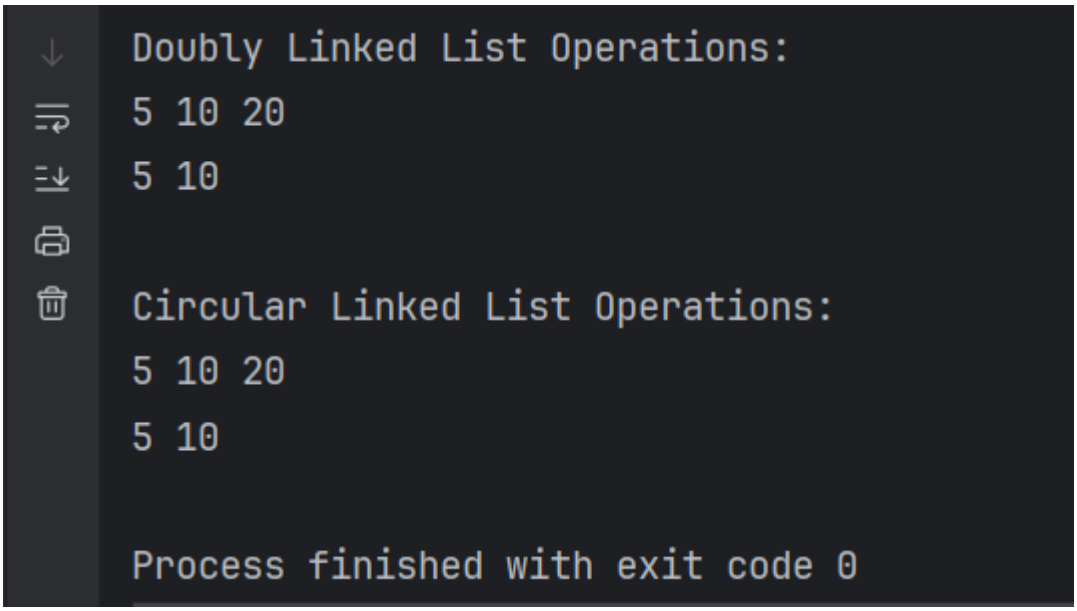
5. Code:

```java
// Node for Doubly Linked List
class DoublyNode {  9 usages
    int data;  2 usages
    DoublyNode prev, next;  5 usages
    public DoublyNode(int data) {  2 usages
        this.data = data;
        prev = next = null;
    }
}

// Node for Circular Linked List
class CircularNode {  11 usages
    int data;  2 usages
    CircularNode next;  22 usages
    public CircularNode(int data) {  2 usages
        this.data = data;
        next = null;
    }
}

class LinkedListDemo {

    // ====== DOUBLY LINKED LIST ======
    DoublyNode headDLL;  17 usages

    void insertAtBeginningDLL(int data) {  2 usages
        DoublyNode newNode = new DoublyNode(data);
        if (headDLL != null) {
            newNode.next = headDLL;
            headDLL.prev = newNode;
        }
        headDLL = newNode;
    }

    void insertAtEndDLL(int data) {  1 usage
        DoublyNode newNode = new DoublyNode(data);
        if (headDLL == null) {
            headDLL = newNode;
            return;
        }
        DoublyNode temp = headDLL;
        while (temp.next != null) temp = temp.next;
        temp.next = newNode;
        newNode.prev = temp;
    }
```

```java
    void deleteAtBeginningDLL() {  no usages
        if (headDLL == null) return;
        headDLL = headDLL.next;
        if (headDLL != null) headDLL.prev = null;
    }

    void deleteAtEndDLL() {  1 usage
        if (headDLL == null) return;
        if (headDLL.next == null) {
            headDLL = null;
            return;
        }
        DoublyNode temp = headDLL;
        while (temp.next != null) temp = temp.next;
        temp.prev.next = null;
    }

    void displayDLL() {  2 usages
        DoublyNode temp = headDLL;
        while (temp != null) {
            System.out.print(temp.data + " ");
            temp = temp.next;
        }
        System.out.println();
    }
```

```java
// ====== CIRCULAR LINKED LIST ======
CircularNode headCLL;  30 usages

void insertAtBeginningCLL(int data) {  2 usages
    CircularNode newNode = new CircularNode(data);
    if (headCLL == null) {
        newNode.next = newNode;
        headCLL = newNode;
        return;
    }
    CircularNode temp = headCLL;
    while (temp.next != headCLL) temp = temp.next;
    newNode.next = headCLL;
    temp.next = newNode;
    headCLL = newNode;
}

void insertAtEndCLL(int data) {  1 usage
    CircularNode newNode = new CircularNode(data);
    if (headCLL == null) {
        newNode.next = newNode;
        headCLL = newNode;
        return;
    }
    CircularNode temp = headCLL;
    while (temp.next != headCLL) temp = temp.next;
    temp.next = newNode;
    newNode.next = headCLL;
}

void deleteAtBeginningCLL() {  no usages
    if (headCLL == null) return;
    if (headCLL.next == headCLL) {
        headCLL = null;
        return;
    }
    CircularNode temp = headCLL;
    while (temp.next != headCLL) temp = temp.next;
    headCLL = headCLL.next;
    temp.next = headCLL;
}

void deleteAtEndCLL() {  1 usage
    if (headCLL == null) return;
    if (headCLL.next == headCLL) {
        headCLL = null;
        return;
    }
    CircularNode temp = headCLL;
    while (temp.next.next != headCLL) temp = temp.next;
    temp.next = headCLL;
}

void displayCLL() {  2 usages
    if (headCLL == null) return;
    CircularNode temp = headCLL;
    do {
        System.out.print(temp.data + " ");
        temp = temp.next;
    } while (temp != headCLL);
    System.out.println();
}

public static void main(String[] args) {
    LinkedListDemo list = new LinkedListDemo();

    System.out.println("Doubly Linked List Operations:");
    list.insertAtBeginningDLL( data: 10);
    list.insertAtEndDLL( data: 20);
    list.insertAtBeginningDLL( data: 5);
    list.displayDLL();
    list.deleteAtEndDLL();
    list.displayDLL();

    System.out.println("\nCircular Linked List Operations:");
    list.insertAtBeginningCLL( data: 10);
    list.insertAtEndCLL( data: 20);
    list.insertAtBeginningCLL( data: 5);
    list.displayCLL();
    list.deleteAtEndCLL();
    list.displayCLL();
}
```

6. Output:


Doubly Linked List Operations:
5 10 20
5 10
Circular Linked List Operations:
5 10 20
5 10
Process finished with exit code 0