



DEPARTMENT OF

Discover. Learn. Empower.

COMPUTER SCIENCE & ENGINEERING

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 attend 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:

```
2
3 // Node for Doubly Linked List
4 class DoublyNode { 9 usages
5     int data; 2 usages
6     DoublyNode prev, next; 5 usages
7     public DoublyNode(int data) { 2 usages
8         this.data = data;
9         prev = next = null;
10    }
11 }
12
13 // Node for Circular Linked List
14 class CircularNode { 11 usages
15     int data; 2 usages
16     CircularNode next; 22 usages
17     public CircularNode(int data) { 2 usages
18         this.data = data;
19         next = null;
20    }
21 }
22
23 class LinkedListDemo {
24
25     // ===== DOUBLY LINKED LIST =====
26     DoublyNode headDLL; 17 usages
27
28     void insertAtBeginningDLL(int data) { 2 usages
29         DoublyNode newNode = new DoublyNode(data);
30         if (headDLL != null) {
31             newNode.next = headDLL;
32             headDLL.prev = newNode;
33         }
34         headDLL = newNode;
35     }
36
37     void insertAtEndDLL(int data) { 1 usage
38         DoublyNode newNode = new DoublyNode(data);
39         if (headDLL == null) {
40             headDLL = newNode;
41             return;
42         }
43         DoublyNode temp = headDLL;
44         while (temp.next != null) temp = temp.next;
45         temp.next = newNode;
46         newNode.prev = temp;
47     }
}
```

```
48
49 void deleteAtBeginningDLL() { no usages
50     if (headDLL == null) return;
51     headDLL = headDLL.next;
52     if (headDLL != null) headDLL.prev = null;
53 }
54
55 void deleteAtEndDLL() { 1 usage
56     if (headDLL == null) return;
57     if (headDLL.next == null) {
58         headDLL = null;
59         return;
60     }
61     DoublyNode temp = headDLL;
62     while (temp.next != null) temp = temp.next;
63     temp.prev.next = null;
64 }
65
66 void displayDLL() { 2 usages
67     DoublyNode temp = headDLL;
68     while (temp != null) {
69         System.out.print(temp.data + " ");
70         temp = temp.next;
71     }
72     System.out.println();
73 }
74 }
```



DEPARTMENT OF

Discover. Learn. Empower.

COMPUTER SCIENCE & ENGINEERING

```
75 // ===== CIRCULAR LINKED LIST =====
76 CircularNode headCLL; 30 usages
77
78 void insertAtBeginningCLL(int data) { 2 usages
79     CircularNode newNode = new CircularNode(data);
80     if (headCLL == null) {
81         newNode.next = newNode;
82         headCLL = newNode;
83         return;
84     }
85     CircularNode temp = headCLL;
86     while (temp.next != headCLL) temp = temp.next;
87     newNode.next = headCLL;
88     temp.next = newNode;
89     headCLL = newNode;
90 }
91
92 void insertAtEndCLL(int data) { 1 usage
93     CircularNode newNode = new CircularNode(data);
94     if (headCLL == null) {
95         newNode.next = newNode;
96         headCLL = newNode;
97         return;
98     }
99     CircularNode temp = headCLL;
100     while (temp.next != headCLL) temp = temp.next;
101     temp.next = newNode;
102     newNode.next = headCLL;
103 }
104
105 void deleteAtBeginningCLL() { no usages
106     if (headCLL == null) return;
107     if (headCLL.next == headCLL) {
108         headCLL = null;
109         return;
110     }
111     CircularNode temp = headCLL;
112     while (temp.next != headCLL) temp = temp.next;
113     headCLL = headCLL.next;
114     temp.next = headCLL;
115 }
116
```

```
117 void deleteAtEndCLL() { 1 usage
118     if (headCLL == null) return;
119     if (headCLL.next == headCLL) {
120         headCLL = null;
121         return;
122     }
123     CircularNode temp = headCLL;
124     while (temp.next.next != headCLL) temp = temp.next;
125     temp.next = headCLL;
126 }
127
128 void displayCLL() { 2 usages
129     if (headCLL == null) return;
130     CircularNode temp = headCLL;
131     do {
132         System.out.print(temp.data + " ");
133         temp = temp.next;
134     } while (temp != headCLL);
135     System.out.println();
136 }
137
138 public static void main(String[] args) {
139     LinkedListDemo list = new LinkedListDemo();
140
141     System.out.println("Doubly Linked List Operations:");
142     list.insertAtBeginningDLL( data: 10);
143     list.insertAtEndDLL( data: 20);
144     list.insertAtBeginningDLL( data: 5);
145     list.displayDLL();
146     list.deleteAtEndDLL();
147     list.displayDLL();
148
149     System.out.println("\nCircular Linked List Operations:");
150     list.insertAtBeginningCLL( data: 10);
151     list.insertAtEndCLL( data: 20);
152     list.insertAtBeginningCLL( data: 5);
153     list.displayCLL();
154     list.deleteAtEndCLL();
155     list.displayCLL();
156 }
157
```



DEPARTMENT OF

Discover. Learn. Empower.

COMPUTER SCIENCE & ENGINEERING

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
```