

DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

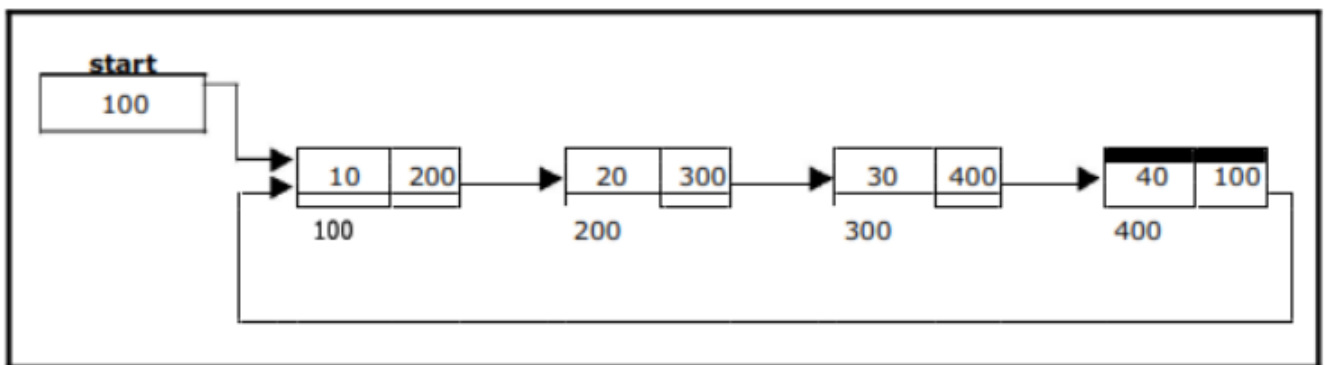
Contents

1. Circular Linked List.....	2
2. Advantages of Circular Linked Lists:	2
4. Creating a Node of Circular LinkedList in JAVA	3
5. Method to Insert a New Node (n) at the Beginning of Circular LinkedList	4
6. Method to Insert a New Node (n) at Last of Circular LinkedList.....	5
7. Method to Insert a New Node (n) Before A Particular Value in a Circular LinkedList	6
8. Method to Delete the First Node in Circular LinkedList.....	7
9. Method to Delete the Last Node in Circular LinkedList	8
10. Display Method for Circular LinkedList.....	9
11. Java Program to InsertAtFirst, InsertAtLast, DeleteFirst & DeleteLast in Circular LinkedList	10

DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

1. Circular Linked List

- ♦ Circular linked list is a linked list where all nodes are connected to form a circle. There is no NULL at the end. A circular linked list can be a singly circular linked list or doubly circular linked list.
- ♦ Circular linked lists are frequently used instead of ordinary linked list because many operations are much easier to implement. In circular linked list no null pointers are used, hence all pointers contain valid address.



2. Advantages of Circular Linked Lists:

1. Any node can be a starting point. We can traverse the whole list by starting from any point. We just need to stop when the first visited node is visited again.
2. Useful for implementation of queue. Unlike this implementation, we don't need to maintain two pointers for front and rear if we use circular linked list. We can maintain a pointer to the last inserted node and front can always be obtained as next of last.
3. Circular lists are useful in applications to repeatedly go around the list. For example, when multiple applications are running on a PC, it is common for the operating system to put the running applications on a list and then to cycle through them, giving each of them a slice of time to execute, and then making them wait while the CPU is given to another application. It is convenient for the operating

DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

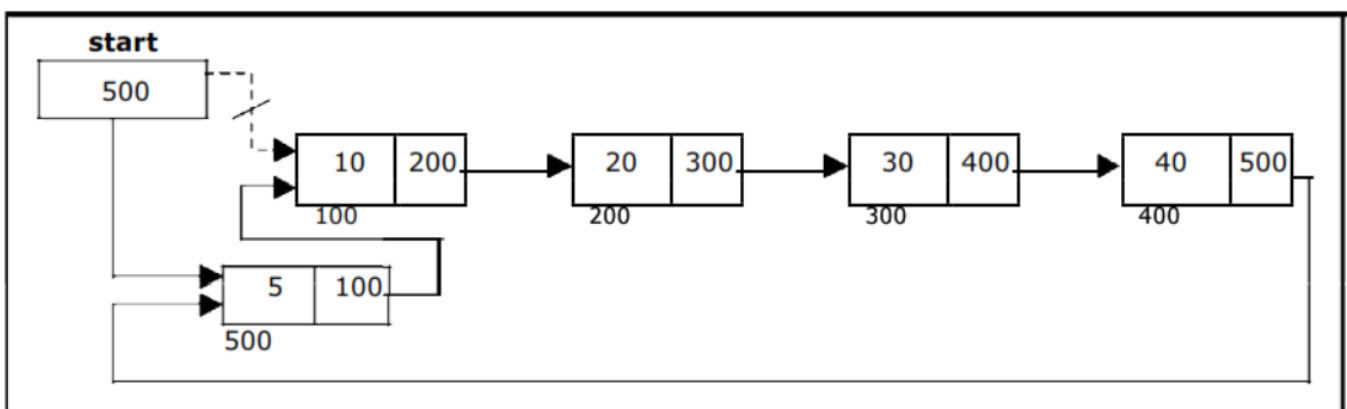
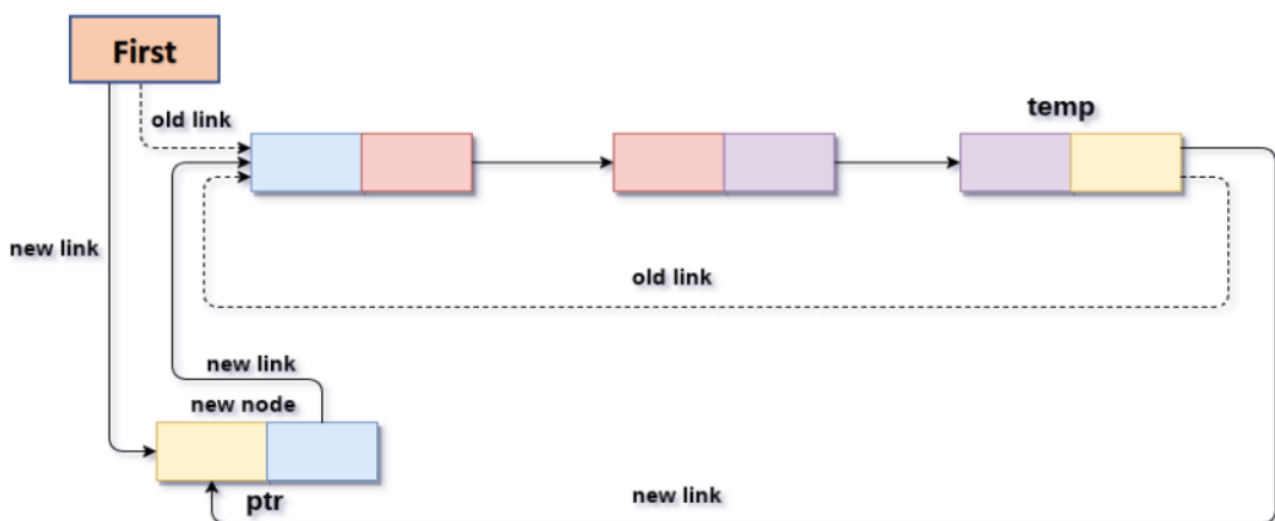
4. Creating a Node of Circular LinkedList in JAVA

```
42 | public static class CircularLinkedList //created a class named Circular LL//
43 | {
44 |     class Node //created a node which contains a data and refrence//
45 |     {
46 |         int data; //the node will have a data of integer type//
47 |         Node next; //it will have a reference variable//
48 |
49 |         Node(int data) //created a constructor to create a node//
50 |         {
51 |             this.data = data; //the value of data will be same as entered by user//
52 |             this.next = null; //the next pointer of the node will initialized with null//
53 |         }
54 |     }
}
```

DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

5. Method to Insert a New Node (n) at the Beginning of Circular LinkedList

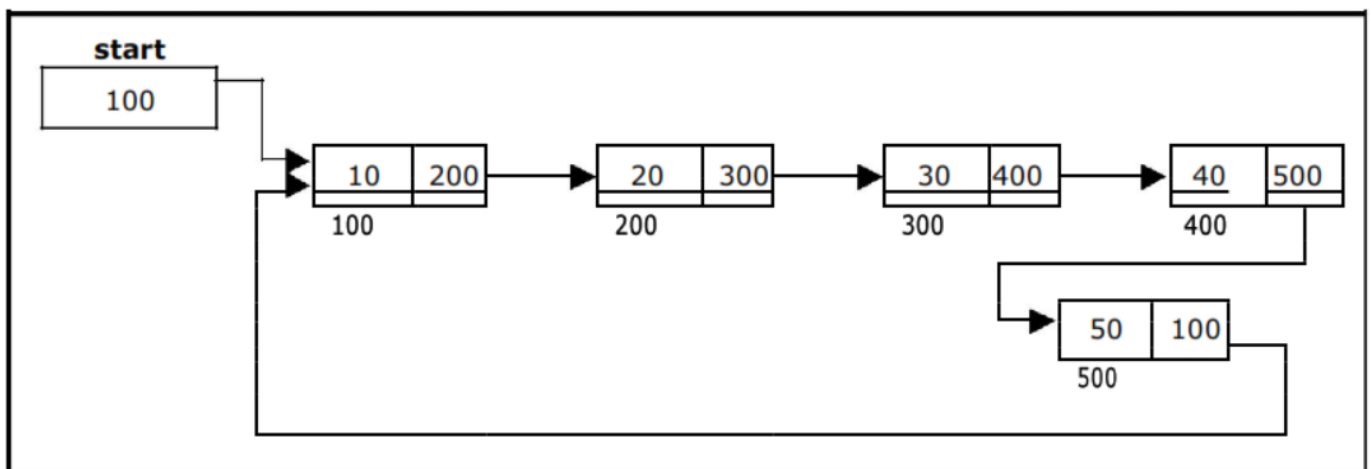
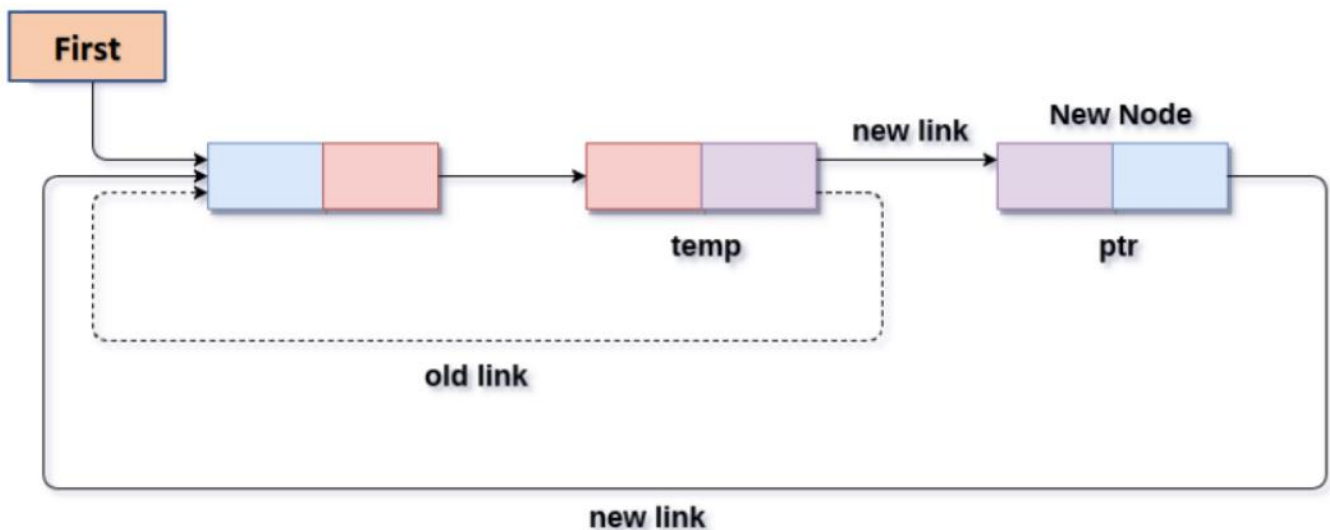
```
56 void insertFirst(int data) //created a method to insert a new node at start//  
57 {  
58     Node n = new Node(data); //In insert method we always required to create a node//  
59     if (first == null) //checking if LinkedList is empty or not//  
60     {  
61         first = n; //if yes then first shall be directly pointing to n//  
62         n.next = n; //and new node's next should be pointing n due to CLL//  
63     }  
64     else //if LinkedList has more values//  
65     {  
66         n.next = first; //new node n's next shall be equal to first//  
67         Node temp = first; //created a temp node with the same value as first//  
68         while (temp.next != first) //until we get the temp = null this loop will run//  
69             temp = temp.next;  
70         temp.next = n; //now temp's next which is showing old first shall be pointing n//  
71         first = n; //and now first shall be pointing towards n//  
72     }  
73     System.out.println(data+" is inserted First");  
74 }
```



DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

6. Method to Insert a New Node (n) at Last of Circular LinkedList

```
75 void insertlast(int data) //created a method to Insert a Node at Last//  
76 {  
77     Node n = new Node(data); //In insert method we always required to create a node//  
78     if (first == null) //checking if LinkedList is empty or not//  
79     {  
80         first = n; //if yes then first shall be directly pointing to n//  
81         n.next = n; //and new node's next should be pointing n due to CLL//  
82     }  
83     else //otherwise//  
84     {  
85         Node temp = first; //created a temp node with the same value as first//  
86         while (temp.next != first) //until we get the temp = null this loop will run//  
87             temp = temp.next;  
88         temp.next = n; //now temp's next which is showing old first shall be pointing n//  
89         n.next = first; //new node n's next shall be equal to first//  
90     }  
91     System.out.println(data+" is inserted Last");  
92 }
```



DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

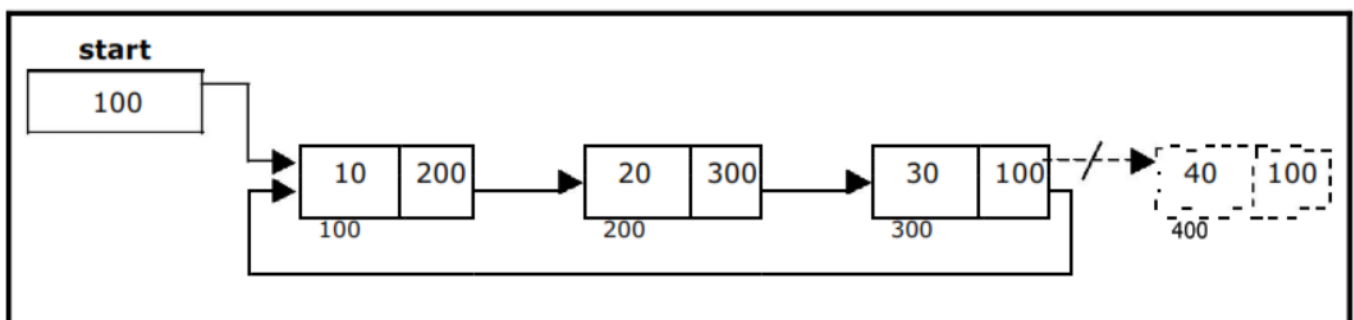
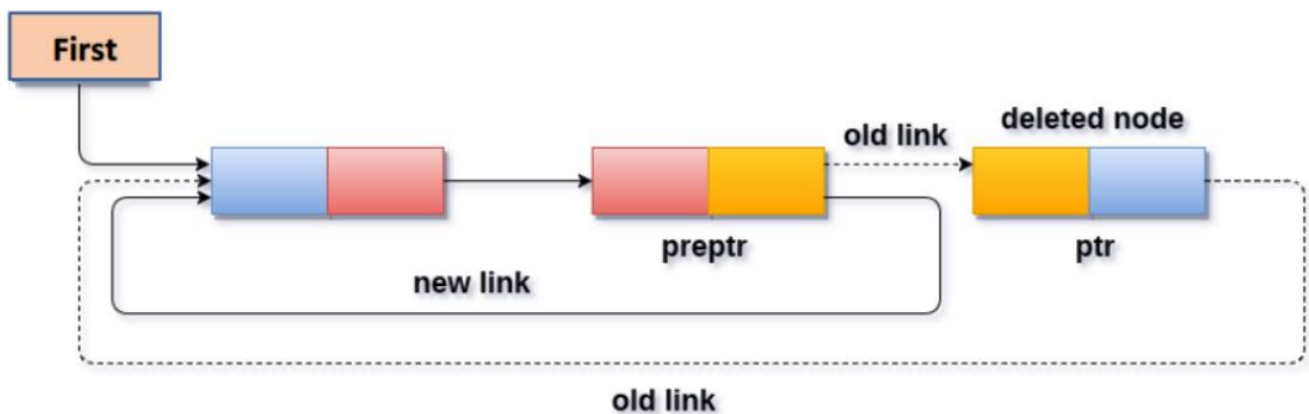
7. Method to Insert a New Node (n) Before A Particular Value in a Circular LinkedList

```
void insertValuebefore(int data,int target) //created a method insert a new node before values//
{
    Node new_node=new Node(data); //In insert method we always required to create a node//
    int flag=0; //used a flag variable to check the value exists in LL or not//
    if(first==null) //checking that LinkedList is empty or not//
    {
        System.out.println("Empty Circular Linked List, So cant insert value before given value ");
    }
    else //otherwise//
    {
        Node temp=first; //created a temp node with the same value as first//
        do
        {
            if(temp.data==target) //if we get temp's data equal to target//
            {
                flag=1; //the flag value will become 1//
            }
            temp=temp.next;
        }while(temp!=first); //untill temp again becomes first in CLL//
        if (flag==1) //now if flag is 1 means value matching//
        {
            if (first.data==target && first.next==first) //checking for 1 node condition//
            {
                new_node.next=first; //new node's next shall be point to first//
                first.next=new_node; //first's next shall be new node//
                first=new_node; //first now can point to new node//
            }
            else if (first.data==target && first.next!=first) //otherwise//
            {
                new_node.next=first; //new node's next equal to first//
                while(temp.next!=first) //until we get the temp's next = null this loop will run//
                {
                    temp=temp.next;
                }
                temp.next=new_node; //temp's next shall be equal to new node//
                first=new_node; //first shall now point to new node//
            }
            else
            {
                //Node temp=first;
                while(temp.next.data!=target)//until we get temp's next's data will be equal to target//
                {
                    temp=temp.next;
                }
                new_node.next=temp.next; // new node's next equal to temp's next//
                temp.next=new_node; //temp's next will be equal to new node//
            }
        }
        System.out.println("Target Value does not exist");
    }
}
```


DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

9. Method to Delete the Last Node in Circular LinkedList

```
117 void dellast() //created a method to Delete a Node at Last//
118 {
119     if (first == null) //checking if LinkedList is empty or not//
120     {
121         System.out.println("UnderFlow");
122     }
123     else if (first.next==first) //checking if LL has only 1 node//
124     {
125         first=null; //then first shall be null//
126     }
127     else
128     {
129         Node temp = first; //created a temp node with the same value as first//
130         while (temp.next.next != first) //until we get the temp's next's next = null this loop will run//
131         {
132             temp = temp.next;
133         }
134         Node del=temp.next; //to release memory of deleted node
135         temp.next = first; //temp's next shall be equal to first//
136         System.out.println(del.data+" is deleted from last");
137         del.next=null;
138     }
139 }
```



DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

10. Display Method for Circular LinkedList

```
240 void display() //created a method to display the data of CircularLinkedList//
241 {
242     if(first == null) //checking if LinkedList is empty or not//
243     {
244         System.out.println("Empty");
245     }
246     else
247     {
248         Node temp = first; //created a temp node with the same value as first//
249         while (temp.next != first) //until we get the temp's next = null this loop will run//
250         {
251             System.out.print(temp.data + "-"); //temp's data get printed//
252             temp = temp.next;
253         }
254         System.out.println(temp.data + " : Circular Linked List"); //printing last node value//
255     }
256 }
```

DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

11. Java Program to InsertAtFirst, InsertAtLast, DeleteFirst & DeleteLast in Circular LinkedList

```
1  class Run7
2  {
3      public static void main(String args[]) {
4          CircularLinkedList L = new CircularLinkedList();
5          L.insertlast(10);
6          L.display();
7          L.insertlast(20);
8          L.display();
9          L.delTargetvalue(20);
10         L.display();
11         L.insertValuebefore(10,10);
12         L.display();
13         L.insertFirst(10);
14         L.display();
15         L.insertFirst(20);
16         L.display();
17         L.insertFirst(30);
18         L.display();
19         L.insertValuebefore(111,20);
20         L.display();
21         L.insertFirst(1);
22         L.display();
23         L.insertFirst(2);
24         L.display();
25         L.insertFirst(3);
26         L.display();
27         L.insertlast(10);
28         L.display();
29         L.insertlast(20);
30         L.display();
31         L.insertlast(30);
32         L.display();
33         L.insertValuebefore(111,30);
34         L.display();
35         L.delTargetvalue(111);
36         L.display();
37         L.dellast();
38         L.display();
39         L.delfirst();
40         L.display();
41     }
42     public static class CircularLinkedList //created a class named Circular LL//
43     {
44         class Node //created a node which contains a data and reference//
45         {
46             int data; //the node will have a data of integer type//
47             Node next; //it will have a reference variable//
48
49             Node(int data) //created a constructor to create a node//
50             {
51                 this.data = data; //the value of data will be same as entered by user//
52                 this.next = null; //the next pointer of the node will initialized with
53                 null//
54             }
55         }
56         Node first = null;
57         void insertFirst(int data) //created a method to insert a new node at start//
58         {
59             Node n = new Node(data); //In insert method we always required to create a
60             node//
61             if (first == null) //checking if LinkedList is empty or not//
62             {
63                 first = n; //if yes then first shall be directly pointing to n//
64                 n.next = n; //and new node's next should be pointing n due to CLL//
65             }
66             else //if LinkedList has more values//
67             {
68                 //...
69             }
70         }
71     }
72 }
```

DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

```
66         n.next = first; //new node n's next shall be equal to first//
67         Node temp = first; //created a temp node with the same value as first//
68         while (temp.next != first) //until we get the temp's next = null this
        loop will run//
69             temp = temp.next;
70             temp.next = n; //now temp's next which is showing old first shall be
        pointing n//
71             first = n; //and now first shall be going towards n//
72     }
73     System.out.println(data+" is inserted First");
74 }
75 void insertlast(int data) //created a method to Insert a Node at Last//
76 {
77     Node n = new Node(data); //In insert method we always required to create a
        node//
78     if (first == null) //checking if LinkedList is empty or not//
79     {
80         first = n; //if yes then first shall be directly pointing to n//
81         n.next = n; //and new node's next should be pointing n due to CLL//
82     }
83     else //otherwise//
84     {
85         Node temp = first; //created a temp node with the same value as first//
86         while (temp.next != first) //until we get the temp's next = null this
        loop will run//
87             temp = temp.next;
88             temp.next = n; //now temp's next which is showing old first shall be
        pointing n//
89             n.next = first; //new node n's next shall be equal to first//
90     }
91     System.out.println(data+" is inserted Last");
92 }
93 void delfirst() //created a method to Delete a Node at Last//
94 {
95     if (first == null) //checking if LinkedList is empty or not//
96     {
97         System.out.println("UnderFlow");
98     }
99     else if (first.next==first) //checking if LL has only 1 node//
100    {
101        first=null; //then first shall be null//
102    }
103    else //otherwise//
104    {
105        Node del = first; //to release memory of deleted node
106        Node temp = first; //created a temp node with the same value as first//
107        while (temp.next != first) //until we get the temp's next = null this
        loop will run//
108        {
109            temp = temp.next;
110        }
111        temp.next = first.next; //then temp's next shall be equal to first's
        next//
112        first = first.next; //first shall be equal to first's next//
113        del.next = null;
114        System.out.println(del.data+" is deleted from first");
115    }
116 }
117 void dellast() //created a method to Delete a Node at Last//
118 {
119     if (first == null) //checking if LinkedList is empty or not//
120     {
121         System.out.println("UnderFlow");
122     }
123     else if (first.next==first) //checking if LL has only 1 node//
124     {
125         first=null; //then first shall be null//
```

DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

```
126     }
127     else
128     {
129         Node temp = first; //created a temp node with the same value as first//
130         while (temp.next.next != first) //until we get the temp's next's next =
            null this loop will run//
131         {
132             temp = temp.next;
133         }
134         Node del=temp.next;//to release memory of deleted node
135         temp.next = first; //temp's next shall be equal to first//
136         System.out.println(del.data+" is deleted from last");
137         del.next=null;
138     }
139 }
140 void insertValuebefore(int data,int target) //created a method insert a new node
    before values//
141 {
142     Node new_node=new Node(data); //In insert method we always required to
        create a node//
143     int flag=0; //used a flag variable to check the value exists in LL or not//
144     if(first==null) //checking that LinkedList is empty or not//
145     {
146         System.out.println("Empty Circular Linked List, So cant insert value
            before given value ");
147     }
148     else //otherwise//
149     {
150         Node temp=first; //created a temp node with the same value as first//
151         do
152         {
153             if(temp.data==target) //if we get temp's data equal to target//
154             {
155                 flag=1; //the flag value will become 1//
156             }
157             temp=temp.next;
158         }while(temp!=first); //untill temp again becomes first in CLL//
159         if (flag==1) //now if flag is 1 means value matching//
160         {
161             if (first.data==target && first.next==first) //checking for 1 node
                condition//
162             {
163                 new_node.next=first; //new node's next shall be point to first//
164                 first.next=new_node; //first's next shall be new node//
165                 first=new_node; //first now can point to new node//
166             }
167             else if (first.data==target && first.next!=first) //otherwise//
168             {
169                 new_node.next=first; //new node's next eual to first//
170                 while(temp.next!=first) //until we get the temp's next = null
                    this loop will run//
171                 {
172                     temp=temp.next;
173                 }
174                 temp.next=new_node; //temp's next shall be equal to new node//
175                 first=new_node; //first shall now point to new node//
176             }
177             else
178             {
179                 //Node temp=first;
180                 while(temp.next.data!=target)//until we get temp's next's data
                    will be equal to target//
181                 {
182                     temp=temp.next;
183                 }
184                 new_node.next=temp.next; // new node's next equal to temp's
                    next//
```

DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

```
185         temp.next=new_node; //temp's next will be equal to new node//
186     }
187 }
188     System.out.println("Target Value does not exist");
189 }
190 }
191 void delTargetvalue(int target) //created a method delete a value//
192 {
193     int flag=0; //used a flag variable to check the value exists in LL or not//
194     if(first==null) //checking that LinkedList is empty or not//
195     {
196         System.out.println("Empty Circular Linked List, So cant insert value
197         before given value ");
198     }
199     else //otherwise//
200     {
201         Node temp=first; //created a temp node with the same value as first//
202         do
203         {
204             if(temp.data==target) //if we get temp's data equal to target//
205             {
206                 flag=1; //the flag value will become 1//
207             }
208             temp=temp.next;
209         }while(temp!=first); //untill temp again becomes first in CLL//
210         if (flag==1) //now if flag is 1 means value matching//
211         {
212             if (first.data==target && first.next==first) //checking for 1 node
213             condition//
214             {
215                 first=null;
216             }
217             else if (first.data==target && first.next!=first) //otherwise//
218             {
219                 Node del=first; //created a new node del equal to first//
220                 while(temp.next!=first) //until we get the temp's next = null
221                 this loop will run//
222                 {
223                     temp=temp.next;
224                 }
225                 temp.next=first.next; //temp's next equal to first's next//
226                 first=first.next; //first shall be now equal to first's next//
227                 del.next=null; //del's next becomes null to break link//
228             }
229             else
230             {
231                 //Node temp=first;
232                 while(temp.next.data!=target) //until we get temp's next's data
233                 will be equal to target//
234                 {
235                     temp=temp.next;
236                 }
237                 temp.next=temp.next.next; //temp's next shall be equal to temp's
238                 next's next//
239             }
240         }
241         System.out.println("Target Value does not exist");
242     }
243 }
244 void display() //created a method to display the data of CircularLinkedList//
245 {
246     if(first == null) //checking if LinkedList is empty or not//
247     {
248         System.out.println("Empty");
249     }
250     else
251     {
252     }
```

DATA STRUCTURES (017013292)
Semester – II
Chapter Name: DOUBLY AND CIRCULAR LINK LIST

```
247         Node temp = first; //created a temp node with the same value as first//
248         while (temp.next != first) //until we get the temp's next = null this
249             loop will run//
250         {
251             System.out.print(temp.data + "-"); //temp's data get printed//
252             temp = temp.next;
253         }
254         System.out.println(temp.data + " : Circular Linked List"); //printing
255         last node value//
256     }
257 }
```