

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

---

## Contents

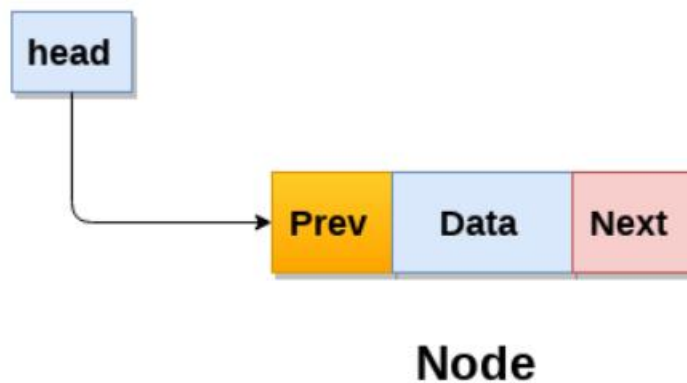
1. Doubly Linked List .....	2
2. Creating a Node of Doubly LinkedList in JAVA .....	3
3. Advantages of Doubly LinkedList over Singly LinkedList .....	3
4. Disadvantages of Doubly LinkedList over Singly LinkedList.....	3
5. Method to Insert a New Node (n) at the Beginning of Doubly LinkedList .....	4
6. Method to Insert a New Node (n) at Last of Doubly LinkedList .....	5
7. Method to Insert a New Node Before Particular Target Value in Doubly Linked List .....	6
8. Method to Delete the First Node in Doubly LinkedList.....	7
9. Method to Delete the Last Node in Doubly LinkedList .....	8
10. Method to Delete a Particular Value in Doubly LinkedList.....	9
11. Display Method for Doubly LinkedList .....	10
12. Java Program to InstertAtFirst, InstertAtLast, DeleteFirst & Deletelast in Doubly LinkedList .....	11

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

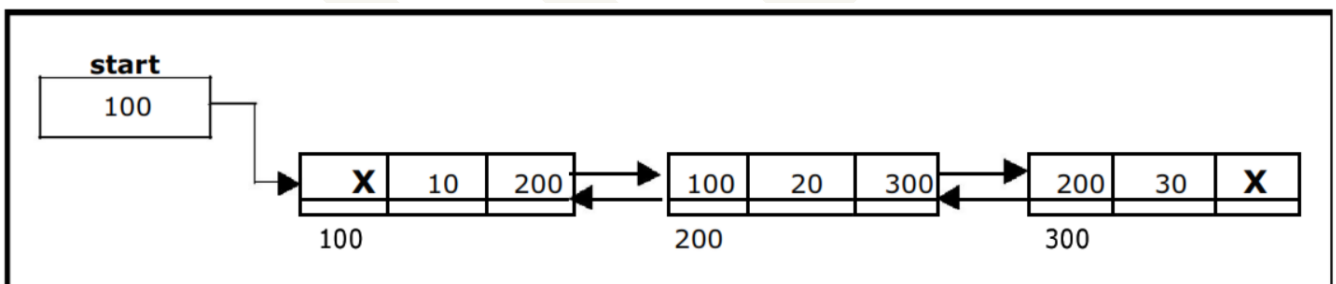
---

## 1. Doubly Linked List

- ♦ Doubly linked list is a complex type of linked list in which a node contains a pointer to the previous as well as the next node in the sequence.
- ♦ Therefore, in a doubly linked list, a node consists of three parts: node data, pointer to the next node in sequence (next pointer), pointer to the previous node (previous pointer). A sample node in a doubly linked list is shown in the figure.



A doubly linked list containing three nodes having numbers having number 10, 20 & 30 in their data part, is shown in the following image.



## 2. Creating a Node of Doubly LinkedList in JAVA

```
2  class DLL //created a class name DLL//
3  {
4      class Node //Created a nested class Node//
5      {
6          int data; //A node is having data of integer type//
7          Node next; //A node is having next reference//
8          Node prev; //A node is having prev reference//
9          Node (int data) //created a constructor with parameter//
10         {
11             this.data=data; //data will the same value entered by user//
12             next=null; //next reference will be initially null//
13             prev=null; //prev reference will be initially null//
14         }
15     }
```

## 3. Advantages of Doubly LinkedList over Singly LinkedList

1. A DLL can be traversed in both forward and backward direction.
2. The delete operation in DLL is more efficient if pointer to the node to be deleted is given.
3. We can quickly insert a new node before a given node.

In singly linked list, to delete a node, pointer to the previous node is needed. To get this previous node, sometimes the list is traversed. In DLL, we can get the previous node using previous pointer.

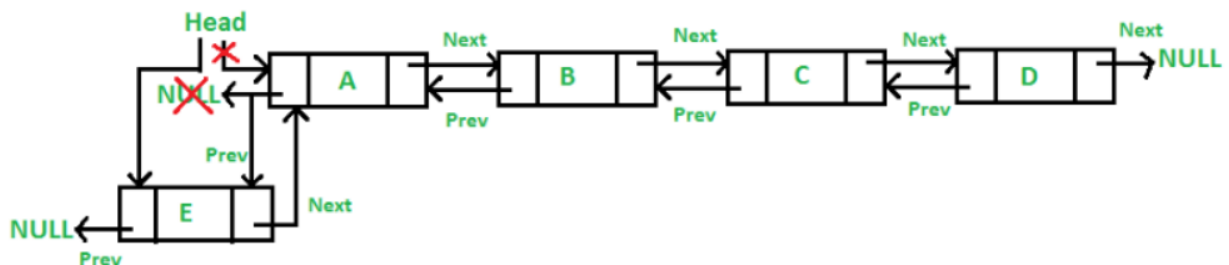
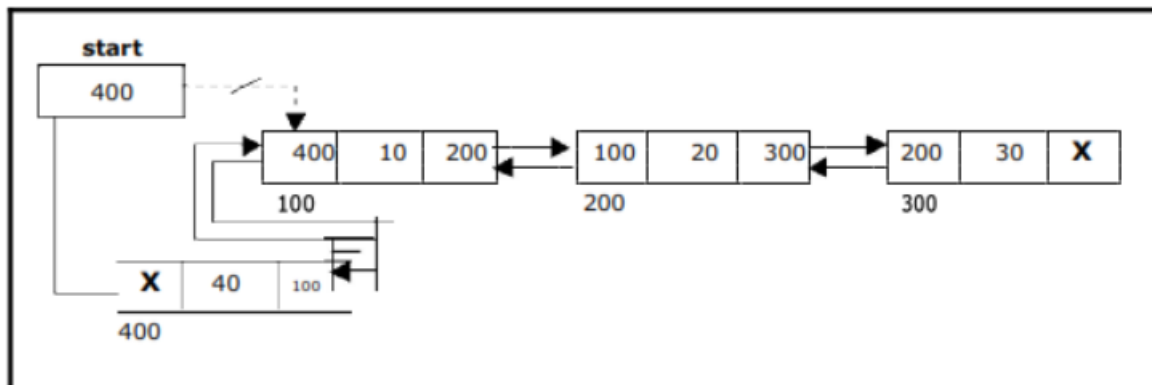
## 4. Disadvantages of Doubly LinkedList over Singly LinkedList

1. Every node of DLL Require extra space for a previous pointer. It is possible to implement DLL with single pointer.
2. All operations require an extra pointer previous to be maintained. For example, in insertion, we need to modify previous pointers together with next pointers. For example in following functions for insertions at different positions, we need 1 or 2 extra steps to set previous pointer.

Node = Prev + Data + Next

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

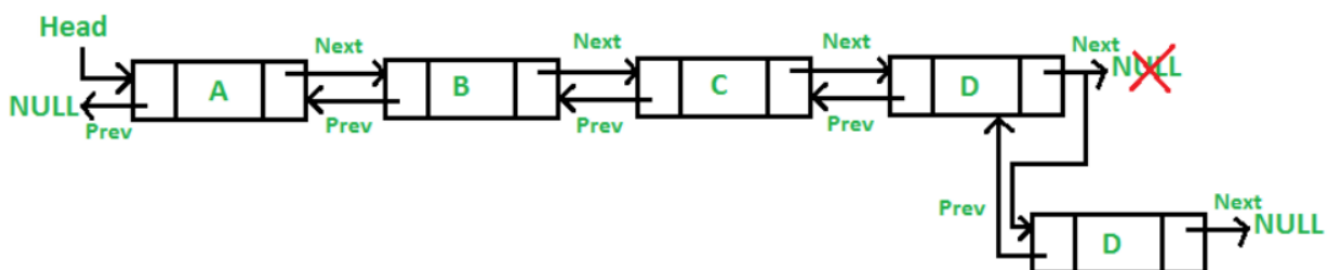
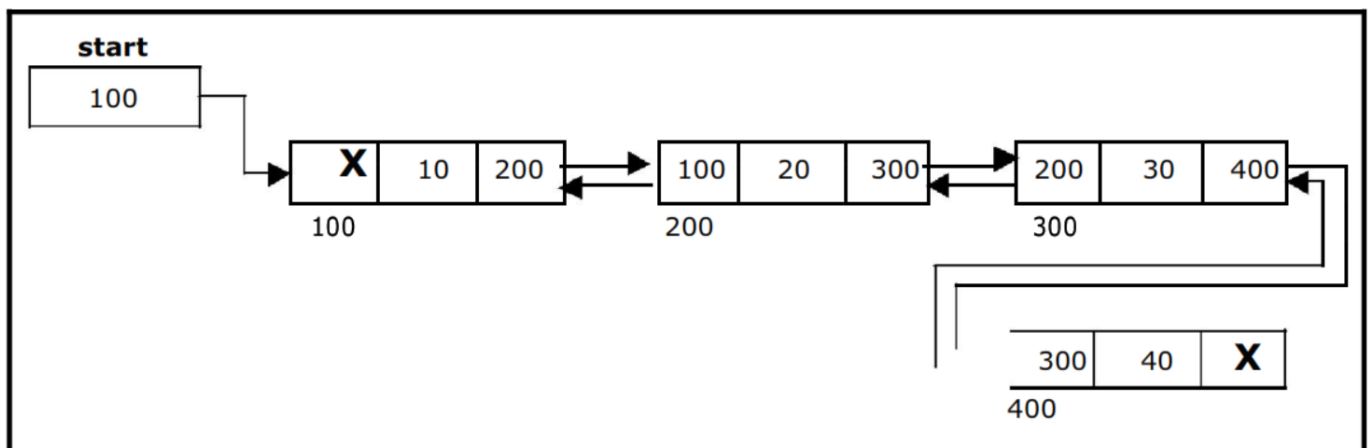
```
35 void insertAtFirst(int data) //created a method to Insert a Node at First//
36 {
37     Node n= new Node(data); //In insert method we always required to create a node//
38     if(first == null) //checking if LinkedList is empty or not//
39     {
40         first =n; //If empty then directly value of n will be assigned to first//
41     }
42     else //If LinkedList has more values then//
43     {
44         n.next=first; //the new node n's next will be set to the first//
45         first.prev=n; //first node's previous will be set to new node n//
46         first=n; //now the first shall point at n//
47     }
48 }
```



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

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

```
49 void insertAtLast(int data) //created a method to Insert a Node at Last//
50 {
51     Node n= new Node(data); //In insert method we always required to create a node//
52     Node temp=first; //created a temp node with the same value as first//
53     if(first == null) //checking if LinkedList is empty or not//
54     {
55         n.prev=null; //In this case new node n's prev shall be null//
56         first=n; //and the first now shall point to the new node n//
57     }
58     else //otherwise//
59     {
60         while(temp.next!=null) //until we get the temp = null this loop will run//
61         {
62             temp=temp.next;
63         }
64         n.prev=temp; //New node n's prev should be equal temp//
65         temp.next=n; //And temp's next shall be equal to new node n//
66     }
67 }
68
```



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

## 7. Method to Insert a New Node Before Particular Target Value in Doubly Linked List

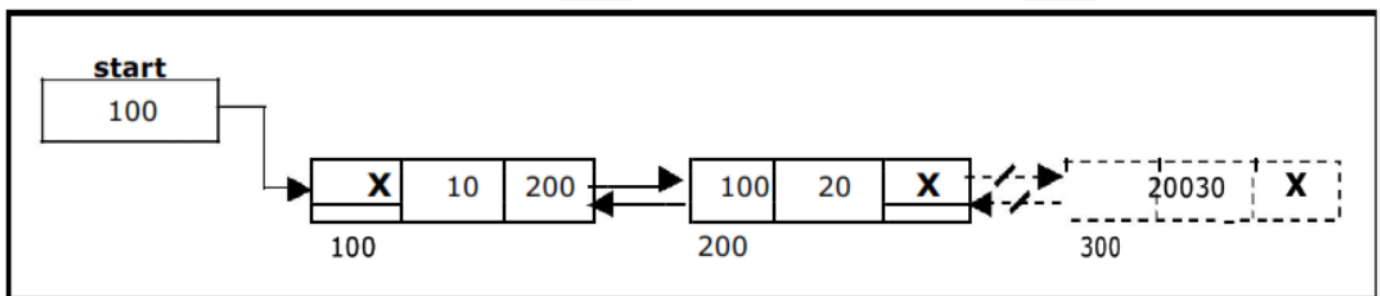
```
109 void InsertBeforeValue(int data, int target)
110 {
111     Node newNode = new Node(data); // Create a new node with the given data
112
113     if (first == null) { // Check if the list is empty
114         System.out.println("LinkedList is Empty");
115         return;
116     }
117
118     if (first.data == target)
119     { // Check if the first node has the target value
120         newNode.next = first; // Set the new node's next to the first node
121         first.prev = newNode; // Set the first node's prev to the new node
122         first = newNode; // Update the first node to be the new node
123         return;
124     }
125     Node current = first.next; // Start from the second node
126     while (current != null)
127     {
128         if (current.data == target)
129         {
130             newNode.next = current; // Set the new node's next to the current node
131             newNode.prev = current.prev; // Set the new node's prev to the current node's prev
132             current.prev.next = newNode; // Update the previous node's next to the new node
133             current.prev = newNode; // Update the current node's prev to the new node
134             return;
135         }
136         current = current.next; // Move to the next node
137     }
138     // If the target value is not found
139     System.out.println("Target value not found in the list");
140 }
```



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

## 9. Method to Delete the Last Node in Doubly LinkedList

```
void deleteLast() //created a method to Delete a Node at Last//
{
    if(first==null) //checking if LinkedList is empty or not//
    {
        System.out.println("Empty");
    }
    else if(first.next==null && first.prev==null) //checking if the LinkedList has only 1 node//
    {
        first=null; //if yes then make first equal to null//
    }
    else //if linked list has more elements then//
    {
        Node temp=first; //created a temp node with the same value as first//
        while(temp.next!=null) //until we get the temp = null this loop will run//
        {
            temp=temp.next; //temp will shift to temp's next//
        }
        temp.prev.next=null; //temp's prev's next will be null//
        temp.prev=null; //and then temp's prev will be also null//
    }
}
```





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

## 10. Method to Delete a Particular Value in Doubly LinkedList

```
143 void deleteValue(int value)
144 {
145     if (first == null)
146     {
147         System.out.println("LINKLIST IS EMPTY");
148     }
149     else if (first.data == value)
150     {
151         System.out.println(first.data);
152         first = first.next;
153         first.prev = null;
154     }
155     else
156     {
157         Node temp = first;
158         while (temp.next != null && temp.data != value) {
159             temp = temp.next;
160         }
161         if (temp.data == value)
162         {
163             if (temp.next != null)
164             {
165                 temp.prev.next = temp.next;
166                 temp.next.prev = temp.prev;
167             }
168             else
169             {
170                 temp.prev.next = null;
171                 temp.prev = null;
172             }
173         }
174         else
175         {
176             System.out.println("VALUE NOT FOUND");
177         }
178     }
179 }
180 }
```

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

---

## 11. Display Method for Doubly LinkedList

```
17 void displayD() //created a method to display the data of DoublyLinkedList//
18 {
19     System.out.print("NULL-"); //As DLL starts with NULL, this will always print NULL at Start//
20     if(first==null) //Checking if LinkedList is Empty or not//
21     {
22         System.out.println("LinkedList is Empty");
23     }
24     else
25     {
26         Node temp = first; //created a temp node with the same value as first//
27         while(temp!=null) //until we get the temp = null this loop will run//
28         {
29             System.out.print(temp.data + "-");
30             temp=temp.next;
31         }
32         System.out.println("NULL"); //each DLL ends with NULL//
33     }
34 }
```

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

---

## 12. Java Program to InsertAtFirst, InsertAtLast, DeleteFirst & DeleteLast in Doubly LinkedList

```
1  import java.util.*;
2  class DLL //created a class name DLL//
3  {
4      class Node //Created a nested class Node//
5      {
6          int data; //A node is having data of integer type//
7          Node next; //A node is having next reference//
8          Node prev; //A node is having prev reference//
9          Node (int data) //created a constructor with parameter//
10         {
11             this.data=data; //data will the same value entered by user//
12             next=null; //next reference will be initially null//
13             prev=null; //prev reference will be initially null//
14         }
15     }
16     Node first=null;
17     void displayD() //created a method to display the data of DoublyLinkedList//
18     {
19         System.out.print("NULL-"); //As DLL starts with NULL, this will always print
20         NULL at Start//
21         if(first==null) //Checking if LinkedList is Empty or not//
22         {
23             System.out.println("LinkedList is Empty");
24         }
25         else
26         {
27             Node temp = first; //created a temp node with the same value as first//
28             while(temp!=null) //until we get the temp = null this loop will run//
29             {
30                 System.out.print(temp.data + "-");
31                 temp=temp.next;
32             }
33             System.out.println("NULL"); //each DLL ends with NULL//
34         }
35     }
36     void insertAtFirst(int data) //created a method to Insert a Node at First//
37     {
38         Node n= new Node(data); //In insert method we always required to create a
39         node//
40         if(first == null) //checking if LinkedList is empty or not//
41         {
42             first =n; //If empty then directly value of n will be assigned to first//
43         }
44         else //If LinkedList has more values then//
45         {
46             n.next=first; //the new node n's next will be set to first//
47             first.prev=n; //first prev will be set to n//
48             first=n; //now the first shall point at n//
49         }
50     }
51     void insertAtLast(int data) //created a method to Insert a Node at Last//
52     {
53         Node n= new Node(data); //In insert method we always required to create a
54         node//
55         Node temp=first; //created a temp node with the same value as first//
56         if(first == null) //checking if LinkedList is empty or not//
57         {
58             n.prev=null; //In this case new node n's prev shall be null//
59             first=n; //and the first now shall point to the new node n//
60         }
61         else //otherwise//
62         {
63             while(temp.next!=null) //until we get the temp = null this loop will
64             run//
65             {
66                 temp=temp.next;
```

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

---

```
64         }
65         n.prev=temp; //New node n's prev should be equal temp//
66         temp.next=n; //And temp's next shall be equal to new node n//
67     }
68 }
69 void deleteFirst() //created a method to Delete a Node at Last//
70 {
71     if(first==null) //checking if LinkedList is empty or not//
72     {
73         System.out.println("Empty");
74     }
75     else if(first.next==null && first.prev==null) //checking if the LinkedList
76     has only 1 node//
77     {
78         first=null; //if yes then make first equal to null//
79     }
80     else //if linked list has more elements then//
81     {
82         Node temp=first; //created a temp node with the same value as first//
83         first=first.next; //shifted the first pointer to first's next//
84         first.prev=null; //make first's prev pointer null//
85         temp.next=null; //make temp(which is at first)'s next pointer null//
86     }
87 }
88 void deleteLast() //created a method to Delete a Node at Last//
89 {
90     if(first==null) //checking if LinkedList is empty or not//
91     {
92         System.out.println("Empty");
93     }
94     else if(first.next==null && first.prev==null) //checking if the LinkedList
95     has only 1 node//
96     {
97         first=null; //if yes then make first equal to null//
98     }
99     else //if linked list has more elements then//
100    {
101        Node temp=first; //created a temp node with the same value as first//
102        while(temp.next!=null) //until we get the temp = null this loop will
103        run//
104        {
105            temp=temp.next; //temp will shift to temp's next//
106        }
107        temp.prev.next=null; //temp's prev's next will be null//
108        temp.prev=null; //and then temp's prev will be also null//
109    }
110 }
111 void InsertBeforeValue(int data, int target)
112 {
113     Node newNode = new Node(data); // Create a new node with the given data
114
115     if (first == null) { // Check if the list is empty
116         System.out.println("LinkedList is Empty");
117         return;
118     }
119
120     if (first.data == target)
121     { // Check if the first node has the target value
122         newNode.next = first; // Set the new node's next to the first node
123         first.prev = newNode; // Set the first node's prev to the new node
124         first = newNode; // Update the first node to be the new node
125         return;
126     }
127     Node current = first.next; // Start from the second node
```

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

---



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

---

```
128         while (current != null)
129         {
130             if (current.data == target)
131             {
132                 newNode.next = current; // Set the new node's next to the current
133                                         // node
134                 newNode.prev = current.prev; // Set the new node's prev to the
135                                             // current node's prev
136                 current.prev.next = newNode; // Update the previous node's next to
137                                             // the new node
138                 current.prev = newNode; // Update the current node's prev to the new
139                                         // node
140                 return;
141             }
142             current = current.next; // Move to the next node
143         }
144         // If the target value is not found
145         System.out.println("Target value not found in the list");
146     }
147     void deleteValue(int value)
148     {
149         if (first == null)
150         {
151             System.out.println("LINKLIST IS EMPTY");
152         }
153         else if (first.data == value)
154         {
155             System.out.println(first.data);
156             first = first.next;
157             first.prev = null;
158         }
159         else
160         {
161             Node temp = first;
162             while (temp.next != null && temp.data != value) {
163                 temp = temp.next;
164             }
165             if (temp.data == value)
166             {
167                 if (temp.next != null)
168                 {
169                     temp.prev.next = temp.next;
170                     temp.next.prev = temp.prev;
171                 }
172                 else
173                 {
174                     temp.prev.next = null;
175                     temp.prev = null;
176                 }
177             }
178             else
179             {
180                 System.out.println("VALUE NOT FOUND");
181             }
182         }
183     }
184 }
185 class Run11 //created the run class to call the method by making objects//
186 {
187     public static void main(String args[])
188     {
189         DLL d = new DLL();
190         d.displayD();
191         d.insertAtFirst(12);
192         d.displayD();
193         d.insertAtLast(22);
194         d.displayD();
195     }
196 }
```

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

---

```
191         d.insertAtFirst(11);
192         d.displayD();
193         d.insertAtLast(222);
194         d.displayD();
195         d.InsertBeforeValue(1,222);
196         d.displayD();
197         d.deleteFirst();
198         d.displayD();
199         d.deleteLast();
200         d.displayD();
201     }
202 }
```