

## DSA\_College\doubly\_linkedList.cpp

```
1  #include<iostream>
2  using namespace std;
3
4  class Node
5  {
6      public :
7      int data ;
8      Node* prev ;
9      Node* next ;
10
11     Node(int data)
12     {
13         this->data = data ;
14         this->prev = NULL ;
15         this->next = NULL ;
16     }
17
18     ~Node()
19     {
20         int value = this->data ;
21         while(this->next != NULL)
22         {
23             delete next ;
24             this->next = NULL ;
25         }
26         cout<<"memory is free for node with data = "<<value<<endl;
27     }
28 };
29
30 int getLength(Node* &head)
31 {
32     Node* temp = head ;
33     int len =0 ;
34     while(temp != NULL)
35     {
36         len++ ;
37         temp = temp->next ;
38     }
39     return len ;
40 }
41
42 void print(Node* &head)
43 {
44     Node* temp = head;
45     while(temp != NULL)
46     {
47         cout<<temp->data<<" ";
48         temp= temp->next ;
49     }
50     cout<<endl;
51 }
```

```
52
53 void insertAtHead(Node* &head, Node* &tail, int d)
54 {
55     if(head == NULL && tail == NULL)
56     {
57         Node* temp = new Node(d) ;
58         head =temp ;
59         tail = temp ;
60     }
61     else{
62         Node* temp = new Node(d) ;
63         temp->next = head ;
64         head->prev = temp ;
65         head = temp ;
66     }
67 }
68
69 void insertAtTail(Node* &tail, Node* &head, int d)
70 {
71     if(head == NULL && tail == NULL)
72     {
73         Node* temp = new Node(d) ;
74         head =temp ;
75         tail = temp ;
76     }
77     else
78     {
79         Node* temp = new Node(d) ;
80         tail->next = temp;
81         temp->prev = tail ;
82         tail = temp ;
83     }
84 }
85
86 void insertAtPosition(Node* &head, Node* &tail, int d, int pos)
87 {
88     if(pos == 1)
89     {
90         insertAtHead(head,tail,d) ;
91         return ;
92     }
93     Node* temp = head ;
94     int cnt = 1 ;
95     while(cnt < pos-1)
96     {
97         temp = temp->next ;
98         cnt++ ;
99     }
100     if(temp->next == NULL)
101     {
102         insertAtTail(tail,head,d) ;
103         return ;
104     }
105     Node* newnode = new Node(d) ;
```

```
106     newnode->next = temp->next ;
107     temp->next->prev = newnode ;
108     temp->next = newnode ;
109     newnode->prev = temp ;
110 }
111
112 void deleteNode(Node* &head, int pos)
113 {
114     if(pos == 1)
115     {
116         Node* temp = head ;
117         temp->next->prev = NULL ;
118         head = head->next ;
119         temp->next = NULL ;
120         delete temp ;
121     }
122     else{
123         Node* prev = NULL ;
124         Node* curr = head ;
125         int cnt = 1 ;
126         while(cnt < pos)
127         {
128             prev = curr ;
129             curr = curr->next ;
130             cnt++ ;
131         }
132         curr->prev = NULL ;
133         prev->next = curr->next ;
134         curr->next->prev = prev ;
135         curr->next = NULL;
136         delete curr ;
137     }
138 }
139
140 int main()
141 {
142     Node* node1 = new Node(10) ; // node1 naam ka object bnaya jisme dat ki value 10 hai
    and prev and next pointers shuru mai null ko point kr rhe
143     Node* head= node1 ; // head naam se ek object bnaya jisme node1 object copy kr diya ya
    fir node1 object se initialize kr diya
144     Node* tail = node1 ;
145     print(head) ; // 10
146
147     insertAtHead(head,tail,11) ; // 11 10
148     print(head) ;
149
150     insertAtTail(tail,head,4) ; // 11 10 4
151     print(head) ;
152
153     insertAtPosition(head,tail,24,2) ; // 11 24 10 4
154     print(head) ;
155
156     deleteNode(head,1) ;
157     print(head) ; // 24 10 4
```

```
158 |  
159 |     return 0 ;  
160 | }
```