

```

1 //Lab Program 9
2 //insert,delete,display
3 #include <stdio.h>
4 #include <stdlib.h>
5
6 struct node{
7     int data;
8     struct node *next,*prev;
9 };
10
11 void insertend(struct node **headptr){
12     struct node *newnode,*temp;
13     int value;
14     printf("Enter value: ");
15     scanf("%d",&value);
16     newnode = (struct node*)malloc(sizeof(struct node));
17     newnode->data = value;
18     newnode->prev = NULL;
19     newnode->next = NULL;
20     if((*headptr) == NULL){
21         (*headptr) = newnode;
22     }
23     else{
24         temp = (*headptr);
25         while(temp->next != NULL)
26             temp = temp->next;
27         temp->next = newnode;
28         newnode->prev = temp;
29     }
30 }
31 void insertbefore(struct node **headptr){
32     struct node *newnode,*temp;
33     int value,ele;
34     printf("Enter value: ");
35     scanf("%d",&value);
36     newnode = (struct node*)malloc(sizeof(struct node));
37     newnode->data = value;
38     newnode->next = NULL;
39     newnode->prev = NULL;
40     if((*headptr) == NULL)
41         (*headptr) = newnode;
42     else{
43         printf("Enter the element before which value is to be inserted: ");
44         scanf("%d",&ele);

```

```

45     if((*headptr)->data == ele){
46         newnode->next = (*headptr);
47         (*headptr)->prev = newnode;
48         (*headptr) = newnode;
49         return;
50     }
51     temp = (*headptr);
52     while(temp->next != NULL){
53         if((temp->next)->data == ele){
54             newnode->next = temp->next;
55             (temp->next)->prev = newnode;
56             temp->next = newnode;
57             newnode->prev = temp;
58             return;
59         }
60         temp = temp->next;
61     }
62     printf("No such element found!\n");
63 }
64 }
65 void deleteend(struct node **headptr){
66     struct node *temp;
67     if((*headptr) == NULL)
68         printf("List is empty\n");
69     else{
70         temp = (*headptr);
71         while((temp->next)->next != NULL)
72             temp = temp->next;
73         temp->next = NULL;
74     }
75 }
76 void deleteval(struct node **headptr){
77     struct node *temp;
78     int value;
79     if((*headptr) == NULL)
80         printf("List is empty\n");
81     else{
82         printf("Enter the value to be deleted: ");
83         scanf("%d",&value);
84         if((*headptr)->data == value){
85             (*headptr) = (*headptr)->next;
86             (*headptr)->prev = NULL;
87             return;
88         }

```

```

90     while((temp->next)->next != NULL)
91         temp = temp->next;
92     if((temp->next)->data == value){
93         temp->next = NULL;
94         return;
95     }
96     temp = (*headptr);
97     while(temp->next != NULL){
98         if((temp->next)->data == value){
99             temp->next = (temp->next)->next;
100             (temp->next)->prev = temp;
101             return;
102         }
103         temp = temp->next;
104     }
105     printf("No such element found!\n");
106 }
107 }
108 void display(struct node *temp){
109     if(temp == NULL)
110         printf("List is empty\n");
111     else{
112         while(temp != NULL){
113             printf("%d\t",temp->data);
114             temp = temp->next;
115         }
116         printf("\n");
117     }
118 }
119 }
120 int main(int argc, char **argv){
121     int choice;
122     struct node *head=NULL;
123     while(choice != 6){
124         printf("Enter choice: 1)insertend 2)insertbefore 3)deleteend 4)deleteval 5)display 6)exit : ");
125         scanf("%d",&choice);
126         switch(choice){
127             case 1:insertend(&head);break;
128             case 2:insertbefore(&head);break;
129             case 3:deleteend(&head);break;
130             case 4:deleteval(&head);break;
131             case 5:display(head);break;
132             case 6:
133                 default:exit(0);

```