


C doublelinkedlist.c >  display(Node *)

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  struct Node {
5      int data;
6      struct Node *prev;
7      struct Node *next;
8  };
9  struct Node* createNode(int data) {
10     struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
11     newNode->data = data;
12     newNode->prev = NULL;
13     newNode->next = NULL;
14     return newNode;
15 }
16 struct Node* createList() {
17     int n, data;
18     struct Node *head = NULL, *temp = NULL, *newNode = NULL;
19
20     printf("Enter number of nodes: ");
21     scanf("%d", &n);
22
23     for(int i = 0; i < n; i++) {
24         printf("Enter data for node %d: ", i + 1);
25         scanf("%d", &data);
26
27         newNode = createNode(data);
28
29         if(head == NULL) {
30             head = newNode;
31             temp = head;
32         } else {
33             temp->next = newNode;
34             newNode->prev = temp;
35             temp = newNode;
36         }
37     }
38     return head;
39 }
40 struct Node* insertLeft(struct Node* head, int val, int newData) {
41     struct Node *temp = head;
42
43     while(temp != NULL && temp->data != val)
44         temp = temp->next;
45
46     if(temp == NULL) {
47         printf("Node with value %d not found!\n", val);
48         return head;
49     }
50 }
```

```

51     struct Node* newNode = createNode(newData);
52
53     if(temp->prev == NULL) {
54         newNode->next = head;
55         head->prev = newNode;
56         head = newNode;
57         return head;
58     }
59     newNode->next = temp;
60     newNode->prev = temp->prev;
61     temp->prev->next = newNode;
62     temp->prev = newNode;
63
64     printf("Inserted %d to the left of %d\n", newData, val);
65     return head;
66 }
67 struct Node* deleteValue(struct Node* head, int val) {
68     struct Node* temp = head;
69
70     while(temp != NULL && temp->data != val)
71         temp = temp->next;
72
73     if(temp == NULL) {
74         printf("Node with value %d not found!\n", val);
75         return head;
76     }
77     if(temp->prev == NULL) {
78         head = temp->next;
79         if(head != NULL)
80             head->prev = NULL;
81         free(temp);
82         printf("Deleted node %d\n", val);
83         return head;
84     }
85     if(temp->next == NULL) {
86         temp->prev->next = NULL;
87         free(temp);
88         printf("Deleted node %d\n", val);
89         return head;
90     }
91     temp->prev->next = temp->next;
92     temp->next->prev = temp->prev;
93     free(temp);
94
95     printf("Deleted node %d\n", val);
96     return head;
97 }

```

```

98 void display(struct Node* head) {
99     struct Node* temp = head;
100     printf("List: ");
101     while(temp != NULL) {
102         printf("%d <-> ", temp->data);
103         temp = temp->next;
104     }
105     printf("NULL\n");
106 }
107 int main() {
108     struct Node* head = NULL;
109     int choice, val, newData;
110
111     while(1) {
112         printf("1.Create List\n2.Insert to the Left of Node\n");
113         printf("3.Delete by Value\n4.Display\n5.Exit\n");
114
115         printf("Enter your choice: ");
116         scanf("%d", &choice);
117
118         switch(choice) {
119             case 1: head = createList();
120                 break;
121             case 2: printf("Enter value of existing node: ");
122                 scanf("%d", &val);
123                 printf("Enter new data to insert: ");
124                 scanf("%d", &newData);
125                 head = insertLeft(head, val, newData);
126                 break;
127             case 3: printf("Enter value to delete: ");
128                 scanf("%d", &val);
129                 head = deleteValue(head, val);
130                 break;
131             case 4: display(head);
132                 break;
133             case 5:
134                 exit(0);
135             default:
136                 printf("Invalid choice!\n");
137         }
138     }
139     return 0;
140 }

```



```
1.Create List
2.Insert to the Left of Node
3.Delete by Value
4.Display
5.Exit
Enter your choice: 1
Enter number of nodes: 4
Enter data for node 1: 12
Enter data for node 2: 23
Enter data for node 3: 36
Enter data for node 4: 65
1.Create List
2.Insert to the Left of Node
3.Delete by Value
4.Display
5.Exit
Enter your choice: 4
List: 12 <-> 23 <-> 36 <-> 65 <-> NULL
1.Create List
2.Insert to the Left of Node
3.Delete by Value
4.Display
5.Exit
Enter your choice: 2
Enter value of existing node: 36
Enter new data to insert: 85
Inserted 85 to the left of 36
1.Create List
2.Insert to the Left of Node
3.Delete by Value
4.Display
5.Exit
Enter your choice: 4
List: 12 <-> 23 <-> 85 <-> 36 <-> 65 <-> NULL
1.Create List
2.Insert to the Left of Node
3.Delete by Value
4.Display
5.Exit
Enter your choice: 3
Enter value to delete: 12
Deleted node 12
1.Create List
2.Insert to the Left of Node
3.Delete by Value
4.Display
5.Exit
Enter your choice: 4
List: 23 <-> 85 <-> 36 <-> 65 <-> NULL
1.Create List
2.Insert to the Left of Node
3.Delete by Value
4.Display
5.Exit
Enter your choice: 5
PS C:\Users\gsm22\OneDrive\Documents\DS>
```