

Doubly-LinkedList(LAB-7) [29/11/25]

code:

```
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
10 struct node *head = NULL;
11
12 struct node* createNode(int data) {
13     struct node *newNode = (struct node*)malloc(sizeof(struct node));
14     newNode->data = data;
15     newNode->prev = NULL;
16     newNode->next = NULL;
17     return newNode;
18 }
19
20 void createList(int n) {
21     int data;
22     struct node *temp, *newNode;
23
24     for (int i = 0; i < n; i++) {
25         printf("Enter data: ");
26         scanf("%d", &data);
27         newNode = createNode(data);
28
29         if (head == NULL) {
30             head = newNode;
31         } else {
32             temp = head;
33             while (temp->next != NULL)
34                 temp = temp->next;
35             temp->next = newNode;
36             newNode->prev = temp;
37         }
38     }
39 }
```

```
40
41     void insertLeft(int key, int data) {
42         struct node *temp = head;
43
44         while (temp != NULL && temp->data != key)
45             temp = temp->next;
46
47         if (temp == NULL) {
48             printf("Node with value %d not found\n", key);
49             return;
50         }
51
52         struct node *newNode = createNode(data);
53         newNode->next = temp;
54         newNode->prev = temp->prev;
55
56         if (temp->prev != NULL)
57             temp->prev->next = newNode;
58         else
59             head = newNode;
60
61         temp->prev = newNode;
62     }
63
64     void deleteNode(int key) {
65         struct node *temp = head;
66
67         while (temp != NULL && temp->data != key)
68             temp = temp->next;
69
70         if (temp == NULL) {
71             printf("Node with value %d not found\n", key);
72             return;
73         }
74
75         if (temp->prev != NULL)
76             temp->prev->next = temp->next;
77         else
78             head = temp->next;
79
80         if (temp->next != NULL)
81             temp->next->prev = temp->prev;
```

```
82             free(temp);
83     }
84
85
86     void display() {
87         struct node *temp = head;
88         if (head == NULL) {
89             printf("List is empty\n");
90             return;
91         }
92         printf("Doubly Linked List: ");
93         while (temp != NULL) {
94             printf("%d <-> ", temp->data);
95             temp = temp->next;
96         }
97         printf("NULL\n");
98     }
99
100    int main() {
101        int choice, n, key, data;
102
103        while (1) {
104            printf("\n1.Create List\n2.Insert Left\n3.Delete\n4.Display\n5.Exit\n");
105            printf("Enter choice: ");
106            scanf("%d", &choice);
107
108            switch (choice) {
109                case 1:
110                    printf("Enter number of nodes: ");
111                    scanf("%d", &n);
112                    createList(n);
113                    break;
114                case 2:
115                    printf("Enter key value: ");
116                    scanf("%d", &key);
117                    printf("Enter new data: ");
118                    scanf("%d", &data);
119                    insertLeft(key, data);
120                    break;
121                case 3:
122                    printf("Enter value to delete: ");
123                    scanf("%d", &key);
124                    deleteNode(key);
```

```
124         deleteNode(key);
125         break;
126     case 4:
127         display();
128         break;
129     case 5:
130         exit(0);
131     default:
132         printf("Invalid choice\n");
133     }
134 }
135 return 0;
136 }
```

Output:

```
1.Create List
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 1
Enter number of nodes: 3
Enter data: 5
Enter data: 7
Enter data: 9

1.Create List
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 2
Enter key value: 3
Enter new data: 1
Node with value 3 not found

1.Create List
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 3
Enter value to delete: 9
```

```
1.Create List
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 2
Enter key value: 7
Enter new data: 10

1.Create List
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 4
Doubly Linked List: 5 <-> 10 <-> 7 <-> NULL

1.Create List
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 5

Process returned 0 (0x0)  execution time : 47.781 s
Press any key to continue.
```