```c
#include <stdio.h>
#include <stdlib.h>
struct Node {
    int data;
    struct Node* prev;
    struct Node* next;
};
struct Node* createNode(int data) {
    struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
    newNode->data = data;
    newNode->prev = NULL;
    newNode->next = NULL;
    return newNode;
}
void insertEnd(struct Node** head, int data) {
    struct Node* newNode = createNode(data);
    if (*head == NULL) {
        *head = newNode;
        return;
    }
    struct Node* temp = *head;
    while (temp->next != NULL)
        temp = temp->next;

    temp->next = newNode;
    newNode->prev = temp;
}
void insertLeft(struct Node** head, int value, int newData) {
    struct Node* newNode = createNode(newData);
    struct Node* temp = *head;
    while (temp != NULL && temp->data != value)
        temp = temp->next;
    if (temp == NULL) {
        printf("Value %d not found in the list!\n", value);
        free(newNode);
        return;
    }
    if (temp->prev == NULL) {
        newNode->next = temp;
        temp->prev = newNode;
        *head = newNode;
        return;
    }
    newNode->prev = temp->prev;
    newNode->next = temp;
    temp->prev->next = newNode;
    temp->prev = newNode;
}
void deleteValue(struct Node** head, int value) {
    struct Node* temp = *head;
    while (temp != NULL && temp->data != value)
```

```
 43         }
 44         newNode->prev = temp->prev;
 45         newNode->next = temp;
 46         temp->prev->next = newNode;
 47         temp->prev = newNode;
 48    }
 49 void deleteValue(struct Node** head, int value) {
 50         struct Node* temp = *head;
 51         while (temp != NULL && temp->data != value)
 52             temp = temp->next;
 53         if (temp == NULL) {
 54             printf("Value %d not found in the list!\n", value);
 55             return;
 56         }
 57         if (temp->prev != NULL)
 58             temp->prev->next = temp->next;
 59         else
 60             *head = temp->next; |
 61
 62         if (temp->next != NULL)
 63             temp->next->prev = temp->prev;
 64         free(temp);
 65         printf("Node with value %d deleted.\n", value);
 66    }
 67 void display(struct Node* head) {
 68         struct Node* temp = head;
 69         printf("List: ");
 70         while (temp != NULL) {
 71             printf("%d <-> ", temp->data);
 72             temp = temp->next;
 73         }
 74         printf("NULL\n");
 75    }
 76 int main() {
 77         struct Node* head = NULL;
 78         int choice, value, newValue;
 79
 80         while (1) {
 81             printf("\n--- Doubly Linked List Menu ---\n");
 82             printf("1. Insert at End\n");
 83             printf("2. Insert to the Left of Node\n");
 84             printf("3. Delete Node by Value\n");
 85             printf("4. Display List\n");
 86             printf("5. Exit\n");
 87             printf("Enter choice: ");
 88             scanf("%d", &choice);
 89
 90             switch (choice) {
 91                 case 1:
 92                     printf("Enter value to insert: ");
 93                     scanf("%d", &value);
```

```c
    }
int main() {
    struct Node* head = NULL;
    int choice, value, newValue;

    while (1) {
        printf("\n--- Doubly Linked List Menu ---\n");
        printf("1. Insert at End\n");
        printf("2. Insert to the Left of Node\n");
        printf("3. Delete Node by Value\n");
        printf("4. Display List\n");
        printf("5. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter value to insert: ");
                scanf("%d", &value);
                insertEnd(&head, value);
                break;

            case 2:
                printf("Enter value of node to insert left of: ");
                scanf("%d", &value);
                printf("Enter new value: ");
                scanf("%d", &newValue);
                insertLeft(&head, value, newValue);
                break;

            case 3:
                printf("Enter value to delete: ");
                scanf("%d", &value);
                deleteValue(&head, value);
                break;

            case 4:
                display(head);
                break;

            case 5:
                exit(0);

            default:
                printf("Invalid choice!\n");
        }
    }

    return 0;
}
```

```
--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 1
Enter value to insert: 10

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 1
Enter value to insert: 20

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 1
Enter value to insert: 30

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 2
Enter value of node to insert left of: 20
Enter new value: 50

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 4
List: 10 <-> 50 <-> 20 <-> 30 <-> NULL

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 3
Enter value to delete: 20
Node with value 20 deleted.

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
```

```
5. Exit
Enter choice: 1
Enter value to insert: 30

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 2
Enter value of node to insert left of: 20
Enter new value: 50

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 4
List: 10 <-> 50 <-> 20 <-> 30 <-> NULL

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 3
Enter value to delete: 20
Node with value 20 deleted.

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 4
List: 10 <-> 50 <-> 30 <-> NULL

--- Doubly Linked List Menu ---
1. Insert at End
2. Insert to the Left of Node
3. Delete Node by Value
4. Display List
5. Exit
Enter choice: 5

Process returned 0 (0x0)   execution time : 81.922 s
Press any key to continue.
```

Activate Windows
Go to Settings to activate Windows.