```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 create(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* temp = *head;
```

```c
void insertLeft(struct Node** head, int value, int 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);
        return;
    }

    struct Node* newNode = createNode(newData);

    newNode->next = temp;
    newNode->prev = temp->prev;

    if (temp->prev != NULL)
        temp->prev->next = newNode;
    else
        *head = newNode;

    temp->prev = newNode;

    printf("Inserted %d to the LEFT of %d.\n", newData, value);
}

void deleteValue(struct Node** head, int value) {
    struct Node* temp = *head;

    while (temp != NULL && temp->data != value)
        temp = temp->next;

    if (temp == NULL) {
        printf("Value %d not found!\n", value);
        return;
```

```c
    if (temp == NULL) {
        printf("Value %d not found!\n", value);
        return;
    }

    if (temp->prev != NULL)
        temp->prev->next = temp->next;
    else
        *head = temp->next;

    if (temp->next != NULL)
        temp->next->prev = temp->prev;

    free(temp);
    printf("Deleted node with value %d.\n", value);
}

void display(struct Node* head) {
    struct Node* temp = head;
    printf("Doubly Linked List: ");
    while (temp != NULL) {
        printf("%d ", temp->data);
        temp = temp->next;
    }
    printf("\n");
}

int main() {
    struct Node* head = NULL;
    int choice, value, newData;

    while (1) {
        printf("\n----- MENU -----\n");
        printf("1. Create (Insert at end)\n");
        printf("2. Insert to LEFT of value\n");
```

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

    while (1) {
        printf("\n----- MENU -----\n");
        printf("1. Create (Insert at end)\n");
        printf("2. Insert to LEFT of value\n");
        printf("3. Delete node by value\n");
        printf("4. Display\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

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

            case 2:
                printf("Enter value to insert: ");
                scanf("%d", &newData);
                printf("Insert to LEFT of which value? ");
                scanf("%d", &value);
                insertLeft(&head, value, newData);
                break;

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

```
                printf("Enter value to insert: ");
                scanf("%d", &value);
                create(&head, value);
                break;

            case 2:
                printf("Enter value to insert: ");
                scanf("%d", &newData);
                printf("Insert to LEFT of which value? ");
                scanf("%d", &value);
                insertLeft(&head, value, newData);
                break;

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

            case 4:
                display(head);
                break;

            case 5:
                printf("Exiting program...\n");
                exit(0);

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

    return 0;
}
```

Blocks | Search results | Cccc | Build log | Build messages | CppCheck/Vera++ | CppCheck/Vera++ messages | Cscope | Debugger | DoxyBlocks | Fortran info | Close

uble linked.c | C/C++ | Windows (CR+LF) | WINDOWS-1252 | Line 82, Col 1, Pos 1788 | Insert | Read/Write | default

```
--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter choice: 1
Enter value to insert: 4

--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter choice: 1
Enter value to insert: 7

--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter choice: 1
Enter value to insert: 9

--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter choice: 2
Enter value of node to insert LEFT of: 7
Enter new data: 2

--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter choice: 4
Doubly Linked List: 4 <-> 2 <-> 7 <-> 9 <-> NULL

--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
```

```
--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter choice: 1
Enter value to insert: 7

--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter choice: 1
Enter value to insert: 9

--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter choice: 2
Enter value of node to insert LEFT of: 7
Enter new data: 2

--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter choice: 4
Doubly Linked List: 4 <-> 2 <-> 7 <-> 9 <-> NULL

--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter choice: 3
Enter value to delete: 7

--- MENU ---
1. Create (Insert at End)
2. Insert Left of a Node
3. Delete a Node by Value
```