doublelinkedlist.c > display(Node *)

```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;
}
struct Node* createList() {
    int n, data;
    struct Node *head = NULL, *temp = NULL, *newNode = NULL;

    printf("Enter number of nodes: ");
    scanf("%d", &n);

    for(int i = 0; i < n; i++) {
        printf("Enter data for node %d: ", i + 1);
        scanf("%d", &data);

        newNode = createNode(data);

        if(head == NULL) {
            head = newNode;
            temp = head;
        } else {
            temp->next = newNode;
            newNode->prev = temp;
            temp = newNode;
        }
    }
    return head;
}
struct Node* insertLeft(struct Node* head, int val, int newData) {
    struct Node *temp = head;

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

    if(temp == NULL) {
        printf("Node with value %d not found!\n", val);
        return head;
    }
```

```c
    struct Node* newNode = createNode(newData);

    if(temp->prev == NULL) {
        newNode->next = head;
        head->prev = newNode;
        head = newNode;
        return head;
    }
    newNode->next = temp;
    newNode->prev = temp->prev;
    temp->prev->next = newNode;
    temp->prev = newNode;

    printf("Inserted %d to the left of %d\n", newData, val);
    return head;
}
struct Node* deleteValue(struct Node* head, int val) {
    struct Node* temp = head;

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

    if(temp == NULL) {
        printf("Node with value %d not found!\n", val);
        return head;
    }
    if(temp->prev == NULL) {
        head = temp->next;
        if(head != NULL)
            head->prev = NULL;
        free(temp);
        printf("Deleted node %d\n", val);
        return head;
    }
    if(temp->next == NULL) {
        temp->prev->next = NULL;
        free(temp);
        printf("Deleted node %d\n", val);
        return head;
    }
    temp->prev->next = temp->next;
    temp->next->prev = temp->prev;
    free(temp);

    printf("Deleted node %d\n", val);
    return head;
}
```

```c
void display(struct Node* head) {
    struct Node* temp = head;
    printf("List: ");
    while(temp != NULL) {
        printf("%d <-> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}
int main() {
    struct Node* head = NULL;
    int choice, val, newData;

    while(1) {
        printf("1.Create List\n2.Insert to the Left of Node\n");
        printf("3.Delete by Value\n4.Display\n5.Exit\n");

        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch(choice) {
            case 1: head = createList();
                break;
            case 2:printf("Enter value of existing node: ");
                scanf("%d", &val);
                printf("Enter new data to insert: ");
                scanf("%d", &newData);
                head = insertLeft(head, val, newData);
                break;
            case 3:printf("Enter value to delete: ");
                scanf("%d", &val);
                head = deleteValue(head, val);
                break;
            case 4:display(head);
                break;
            case 5:
                exit(0);
            default:
                printf("Invalid choice!\n");
        }
    }
    return 0;
}
```

```
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>
```