```c
#include <stdio.h>
#include <stdlib.h>

typedef struct Node {
    int data;
    struct Node* prev;
    struct Node* next;
} Node;

Node* createNode(int data) {
    Node* newNode = (Node*)malloc(sizeof(Node));
    newNode->data = data;
    newNode->prev = NULL;
    newNode->next = NULL;
    return newNode;
}

Node* createDoublyLinkedList(int n) {
    if (n <= 0) return NULL;

    int data;
    printf("Enter data for node 1: ");
    scanf("%d", &data);

    Node* head = createNode(data);
    Node* temp = head;

    for (int i = 2; i <= n; i++) {
        printf("Enter data for node %d: ", i);
        scanf("%d", &data);

        Node* newNode = createNode(data);
        temp->next = newNode;
        newNode->prev = temp;
        temp = newNode;
    }
    return head;
}


void insertLeft(Node** head, int target, int data) {
    Node* temp = *head;
    while (temp != NULL && temp->data != target) {
        temp = temp->next;
    }

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

    Node* newNode = createNode(data);
    newNode->next = temp;
    newNode->prev = temp->prev;
```

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


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

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

    if (temp == NULL) {
        printf("Node with 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("Node with value %d deleted!\n", value);
}

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


int main() {
    Node* head = NULL;
    int choice, n, target, data, value;

    while (1) {
        printf("\nMenu:\n");
```

```
        printf("Node with value %d not found!\n", target);
        return;
    }

    Node* newNode = createNode(data);
    newNode->next = temp;
    newNode->prev = temp->prev;

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


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

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

    if (temp == NULL) {
        printf("Node with 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("Node with value %d deleted!\n", value);
}

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

```
        temp = temp->next;
    }
    printf("\n");
}int main() {
    Node* head = NULL;
    int choice, n, target, data, value;
     while (1) {
        printf("\nMenu:\n");
        printf("1. Create Doubly Linked List\n");
        printf("2. Insert a Node to the Left of a Specific Node\n");
        printf("3. Delete a Node by Value\n");
        printf("4. Display List\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter the number of nodes: ");
                scanf("%d", &n);
                head = createDoublyLinkedList(n);
                break;
                case 2:
                if (head == NULL) {
                    printf("List is empty! Create a list first.\n");
                } else {
                    printf("Enter the target value: ");
                    scanf("%d", &target);
                    printf("Enter the value to insert: ");
                    scanf("%d", &data);
                    insertLeft(&head, target, data);
                }
                break;case 3:
                if (head == NULL) {
                    printf("List is empty! Create a list first.\n");
                } else {
                    printf("Enter the value to delete: ");
                    scanf("%d", &value);
                    deleteNode(&head, value);
                }
                break;case 4:
                if (head == NULL) {
                    printf("List is empty!\n");
                } else {
                    displayList(head);
                }
                break;case 5:
                printf("Exiting...\n");
                exit(0);default:
                printf("Invalid choice! Please try again.\n");}
    }return 0;
}
```

```
Menu:
1. Create Doubly Linked List
2. Insert a Node to the Left of a Specific Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter your choice: 1
Enter the number of nodes: 3
Enter data for node 1: 13
Enter data for node 2: 14
Enter data for node 3: 15

Menu:
1. Create Doubly Linked List
2. Insert a Node to the Left of a Specific Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter your choice: 2
Enter the target value: 14
Enter the value to insert: 25

Menu:
1. Create Doubly Linked List
2. Insert a Node to the Left of a Specific Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter your choice: 4
Doubly Linked List: 13 25 14 15

Menu:
1. Create Doubly Linked List
2. Insert a Node to the Left of a Specific Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter your choice: 3
Enter the value to delete: 15
Node with value 15 deleted!

Menu:
1. Create Doubly Linked List
2. Insert a Node to the Left of a Specific Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter your choice: 4
Doubly Linked List: 13 25 14
```

```
1. Create Doubly Linked List
2. Insert a Node to the Left of a Specific Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter your choice: 3
Enter the value to delete: 15
Node with value 15 deleted!

Menu:
1. Create Doubly Linked List
2. Insert a Node to the Left of a Specific Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter your choice: 4
Doubly Linked List: 13 25 14

Menu:
1. Create Doubly Linked List
2. Insert a Node to the Left of a Specific Node
3. Delete a Node by Value
4. Display List
5. Exit
Enter your choice: 5
Exiting...

Process returned 0 (0x0)   execution time : 1567.972 s
Press any key to continue.
```