doubly linked list

```
#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 insertLeft(struct Node** head, struct Node* targetNode,
                                    int data)
{
    if (!targetNode)
      {
       printf("Error: Target node is Null\n");
       return;
      }
    struct Node *newNode = createNode(data);
    if(targetNode->prev != NULL)
        targetNode->prev->next = newNode;
    else
        *head = newNode;
    newNode->prev = targetNode->prev;
    newNode->next = targetNode;
    targetNode->prev = newNode;
}
```

```
void deleteNode(struct Node **head, int value)
{
    struct Node *current = *head;
    while (current != NULL)
    {
        if(current->data == value)
        {
            if (current->prev != NULL)
                current->prev->next = current->next;
            else
                *head = current->next;
            if (current->next != NULL)
                current->next->prev = current->prev;

            free(current);
            printf("Node with value %d deleted
                successfully\n", value);

            return;
        }
        current = current->next;
    }
    printf("Node with value %d not found\n", value);
}

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

```
int main()
{
    struct Node* head = Null;
    int choice, data, insertValue, deleteValue;
    do {
        printf("\n Menu :\n");
        printf(" 1. Insert a node \n");
        printf(" 2. Delete a node \n" );
        printf(" 3. Display the list \n");
        printf (" 4. Exit\n" );
        printf(" Enter your choice: ");
        scanf ("%d", &choice);

        switch(choice)
            {
            case 1:
                printf(" Enter data for the new node: ");
                scanf("%d", &data);

                if(head == NULL)
                    head = createNode(data);

                else
                    {
                    struct Node* current = head;
                    while(current->next != NULL)
                        current = current->next;

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

                    }
                break;

            case 2:
                printf(" Enter the value of the node to delete: ");
                scanf(" %d", &deleteValue);

                deleteNode(&head, deleteValue);
                break;

            case 3:
                displayList(head);
                break;
            case 4:
                printf(" Exiting the program\n");
                break;

            default:
                print("Invalid choice");
    }
    } while (choice != 4);
    return 0;
}
```

```
Menu:
1. Insert a node
2. Delete a node
3. Display the list
4. Exit
Enter your choice: 1
Enter data for the new node: 1

Menu:
1. Insert a node
2. Delete a node
3. Display the list
4. Exit
Enter your choice: 1
Enter data for the new node: 2

Menu:
1. Insert a node
2. Delete a node
3. Display the list
4. Exit
Enter your choice: 1
Enter data for the new node: 3

Menu:
1. Insert a node
2. Delete a node
3. Display the list
4. Exit
Enter your choice: 1
Enter data for the new node: 4

Menu:
1. Insert a node
2. Delete a node
3. Display the list
4. Exit
Enter your choice: 3
Doubly Linked List: 1 -> 2 -> 3 -> 4 -> NULL

Menu:
1. Insert a node
2. Delete a node
3. Display the list
4. Exit
Enter your choice: 2
Enter the value of the node to delete: 2
Node with value 2 deleted successfully

Menu:
1. Insert a node
2. Delete a node
3. Display the list
4. Exit
Enter your choice: 3
Doubly Linked List: 1 -> 3 -> 4 -> NULL
```