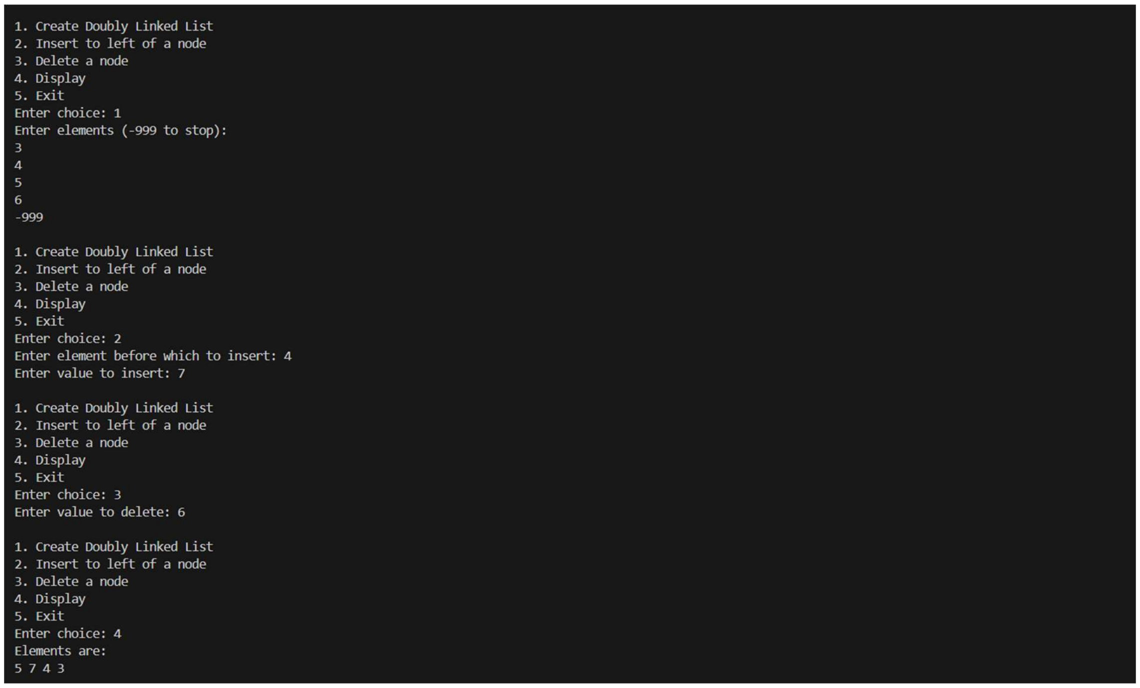
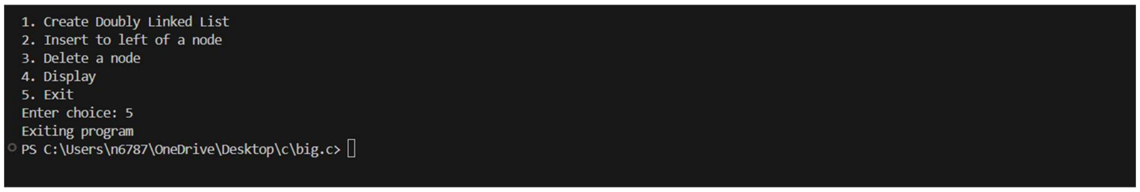
WAP to Implement doubly link list with primitive operations a) Create a doubly linked list. b) Insert a new node to the left of the node. c) Delete the node based on a specific value d) Display the contents of the list

```c
#include <stdio.h>

#include <stdlib.h>


struct node {

    struct node* prev;

    int info;

    struct node* next;

};


struct node* createdoubly() {

    struct node *start = NULL, *p;

    int item;


    printf("Enter elements (-999 to stop):\n");

    scanf("%d", &item);


    while (item != -999) {

        p = (struct node*)malloc(sizeof(struct node));

        p->info = item;

        p->prev = NULL;

        p->next = start;


        if (start != NULL)

            start->prev = p;
        start = p;

        scanf("%d", &item);

    }

    return start;

}


struct node* insertleft(struct node* start, int val, int ele) {

    struct node *p, *temp;


    if (start == NULL) {

        printf("List is empty\n");

        return start;

    }


    temp = start;


    while (temp != NULL && temp->info != ele)

        temp = temp->next;


    if (temp == NULL) {

        printf("Element not found\n");

        return start;

    }


    p = (struct node*)malloc(sizeof(struct node));

    p->info = val;


    p->next = temp;
    p->prev = temp->prev;

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

    temp->prev = p;

    return start;
}

struct node* deleteNode(struct node* start, int val) {
    struct node *temp;

    if (start == NULL) {
        printf("List is empty\n");
        return start;
    }

    temp = start;

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

    if (temp == NULL) {
        printf("Element not found\n");
        return start;
    }

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

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

    free(temp);
    return start;
}

void displaylk(struct node* start) {
    struct node* temp;

    if (start == NULL) {
        printf("List is empty\n");
        return;
    }

    printf("Elements are:\n");
    temp = start;
    while (temp != NULL) {
        printf("%d ", temp->info);
        temp = temp->next;
    }
    printf("\n");
}

int main() {
    struct node* head = NULL;
    int choice, val, ele;

    while (1) {
        printf("\n1. Create Doubly Linked List");
        printf("\n2. Insert to left of a node");
        printf("\n3. Delete a node");
        printf("\n4. Display");
        printf("\n5. Exit");
        printf("\nEnter choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                head = createdoubly();
                break;

            case 2:
                printf("Enter element before which to insert: ");
                scanf("%d", &ele);
                printf("Enter value to insert: ");
                scanf("%d", &val);
                head = insertleft(head, val, ele);
                break;
            case 3:
                printf("Enter value to delete: ");
                scanf("%d", &val);
                head = deleteNode(head, val);
                break;

            case 4:
                displaylk(head);
                break;

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

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

Output:



```
1. Create Doubly Linked List
2. Insert to left of a node
3. Delete a node
4. Display
5. Exit
Enter choice: 1
Enter elements (-999 to stop):
3
4
5
6
-999

1. Create Doubly Linked List
2. Insert to left of a node
3. Delete a node
4. Display
5. Exit
Enter choice: 2
Enter element before which to insert: 4
Enter value to insert: 7

1. Create Doubly Linked List
2. Insert to left of a node
3. Delete a node
4. Display
5. Exit
Enter choice: 3
Enter value to delete: 6

1. Create Doubly Linked List
2. Insert to left of a node
3. Delete a node
4. Display
5. Exit
Enter choice: 4
Elements are:
5 7 4 3
```



```
 1. Create Doubly Linked List
 2. Insert to left of a node
 3. Delete a node
 4. Display
 5. Exit
 Enter choice: 5
 Exiting program
PS C:\Users\n6787\OneDrive\Desktop\c\big.c>
```