7A. 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

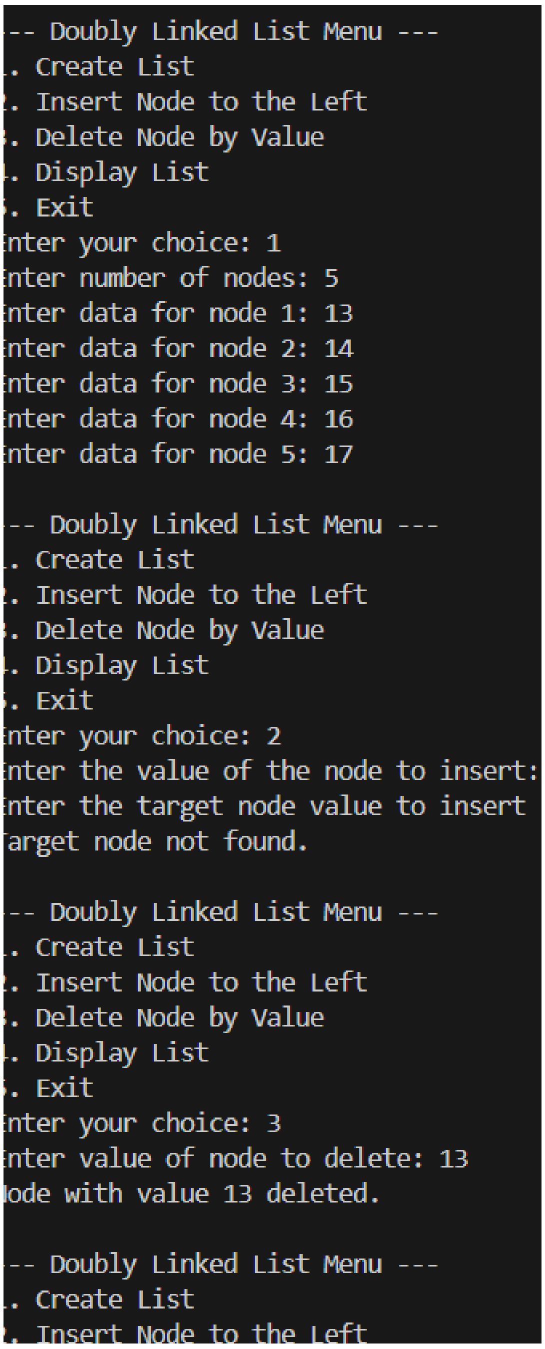
d) Display the contents of the list

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

// Definition of node
struct node {
    int data;
    struct node *prev;
    struct node *next;
};

struct node *head = NULL;

// Create doubly linked list
void create() {
    int n, value;
    struct node *temp = NULL, *newnode;

    printf("Enter number of nodes: ");
    scanf("%d", &n);
    for (int i = 0; i < n; i++) {
        newnode = (struct node *)malloc(sizeof(struct node));
        printf("Enter data for node %d: ", i + 1);
        scanf("%d", &value);

        newnode->data = value;
        newnode->prev = newnode->next = NULL;

        if (head == NULL) {
            head = temp = newnode;
        } else {
            temp->next = newnode;
            newnode->prev = temp;
            temp = newnode;
        }
    }
}

// Insert new node to the left of a given value
void insert_left() {
    int value, target;
    struct node *temp = head, *newnode;

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

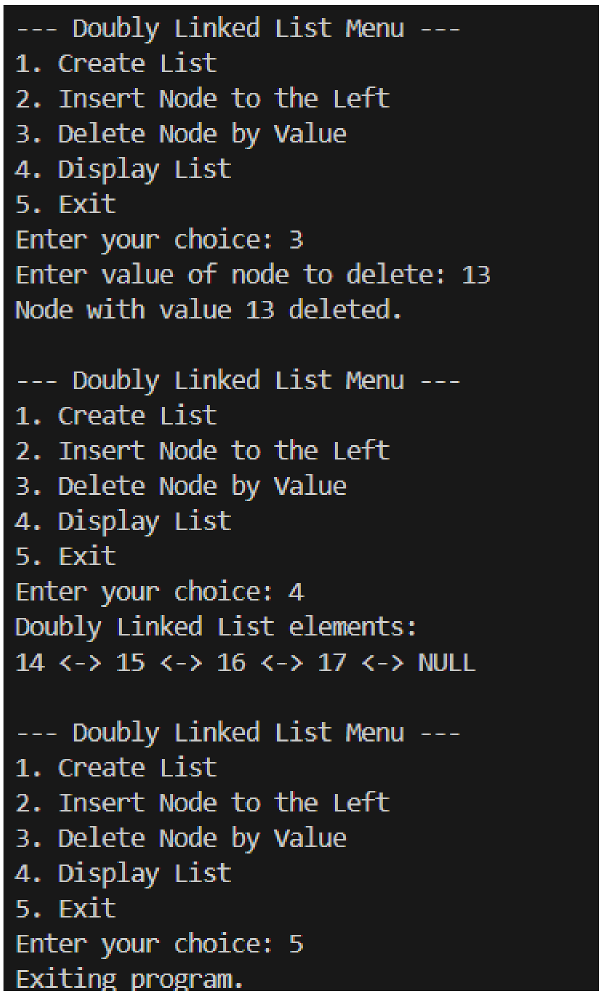
```
printf("Enter the value of the node to insert: ");

scanf("%d", &value);

printf("Enter the target node value to insert before: ");

scanf("%d", &target);


while (temp != NULL && temp->data != target) {

   temp = temp->next;

}


if (temp == NULL) {

   printf("Target node not found.\n");

   return;

}


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

newnode->data = value;


newnode->next = temp;

newnode->prev = temp->prev;


if (temp->prev != NULL)

   temp->prev->next = newnode;

else

   head = newnode; // inserting at beginning


temp->prev = newnode;


printf("%d inserted to the left of %d.\n", value, target);
```

```
}

// Delete node by specific value
void delete_node() {
    int value;
    struct node *temp = head;

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

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

    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; // deleting first node
    if (temp->next != NULL)
        temp->next->prev = temp->prev;

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

// Display doubly linked list
void display() {
    struct node *temp = head;

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

    printf("Doubly Linked List elements:\n");
    while (temp != NULL) {
        printf("%d <-> ", temp->data);
        temp = temp->next;
    }
    printf("NULL\n");
}

// Main function
int main() {
    int choice;
    do {
        printf("\n--- Doubly Linked List Menu ---\n");
        printf("1. Create List\n");
        printf("2. Insert Node to the Left\n");
        printf("3. Delete Node by Value\n");
        printf("4. Display List\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1: create(); break;
            case 2: insert_left(); break;
            case 3: delete_node(); break;
            case 4: display(); break;
            case 5: printf("Exiting program.\n"); break;
            default: printf("Invalid choice! Try again.\n");
        }
    } while (choice != 5);

    return 0;
}
```

OUTPUT:



```
-- Doubly Linked List Menu ---
. Create List
. Insert Node to the Left
. Delete Node by Value
. Display List
. Exit
nter your choice: 1
nter number of nodes: 5
nter data for node 1: 13
nter data for node 2: 14
nter data for node 3: 15
nter data for node 4: 16
nter data for node 5: 17

-- Doubly Linked List Menu ---
. Create List
. Insert Node to the Left
. Delete Node by Value
. Display List
. Exit
nter your choice: 2
nter the value of the node to insert:
nter the target node value to insert
arget node not found.

-- Doubly Linked List Menu ---
. Create List
. Insert Node to the Left
. Delete Node by Value
. Display List
. Exit
nter your choice: 3
nter value of node to delete: 13
ode with value 13 deleted.

-- Doubly Linked List Menu ---
. Create List
. Insert Node to the Left
```



```
--- Doubly Linked List Menu ---
1. Create List
2. Insert Node to the Left
3. Delete Node by Value
4. Display List
5. Exit
Enter your choice: 3
Enter value of node to delete: 13
Node with value 13 deleted.

--- Doubly Linked List Menu ---
1. Create List
2. Insert Node to the Left
3. Delete Node by Value
4. Display List
5. Exit
Enter your choice: 4
Doubly Linked List elements:
14 <-> 15 <-> 16 <-> 17 <-> NULL

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