/* 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 */

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


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


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

```
}


void createList(struct Node** head, int value) {
    struct Node* newNode = createNode(value);

    if (*head == NULL) {
        *head = newNode;
        return;
    }

    struct Node* temp = *head;
    while (temp->next != NULL)
        temp = temp->next;

    temp->next = newNode;
    newNode->prev = temp;
}


void insertLeft(struct Node** head, int value) {
    struct Node* newNode = createNode(value);

    newNode->next = *head;
```

```
    if (*head != NULL)
        (*head)->prev = newNode;

    *head = newNode;
}


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

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

    if (temp == NULL) {
        printf("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 display(struct Node* head) {
    struct Node* temp = head;

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

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

}

int main() {
    struct Node* head = NULL;
    int choice, value;
```

```
while (1) {
    printf("1. Create a node at end)\n");
    printf("2. Insert Left of a node\n");
    printf("3. Delete a value\n");
    printf("4. Display List\n");
    printf("5. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1:
            printf("Enter value: ");
            scanf("%d", &value);
            createList(&head, value);
            break;

        case 2:
            printf("Enter value: ");
            scanf("%d", &value);
            insertLeft(&head, value);
            break;

        case 3:
            printf("Enter value to delete: ");
```

```
            scanf("%d", &value);
            deleteNode(&head, value);
            break;

        case 4:
            display(head);
            break;

        case 5:
            exit(0);

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

   return 0;
}
```

OUTPUT:-


"C:\Users\88nin\OneDrive\Do
1. Create a node at end)
2. Insert Left of a node
3. Delete a value
4. Display List
5. Exit
Enter your choice: 1
Enter value: 60
1. Create a node at end)
2. Insert Left of a node
3. Delete a value
4. Display List
5. Exit
Enter your choice: 1
Enter value: 80
1. Create a node at end)
2. Insert Left of a node
3. Delete a value
4. Display List
5. Exit
Enter your choice: 2
Enter value: 40
1. Create a node at end)
2. Insert Left of a node
3. Delete a value
4. Display List
5. Exit
Enter your choice: 3
Enter value to delete: 60
Node with value 60 deleted.
1. Create a node at end)
2. Insert Left of a node
3. Delete a value
4. Display List
5. Exit
Enter your choice: 4
Doubly Linked List:
40
80


"C:\Users\88nin\OneDrive\Do
Enter value: 60
1. Create a node at end)
2. Insert Left of a node
3. Delete a value
4. Display List
5. Exit
Enter your choice: 1
Enter value: 80
1. Create a node at end)
2. Insert Left of a node
3. Delete a value
4. Display List
5. Exit
Enter your choice: 2
Enter value: 40
1. Create a node at end)
2. Insert Left of a node
3. Delete a value
4. Display List
5. Exit
Enter your choice: 3
Enter value to delete: 60
Node with value 60 deleted.
1. Create a node at end)
2. Insert Left of a node
3. Delete a value
4. Display List
5. Exit
Enter your choice: 4
Doubly Linked List:
40
80
1. Create a node at end)
2. Insert Left of a node
3. Delete a value
4. Display List
5. Exit
Enter your choice: 5
Process returned 0 (0x0) execution time : 54.390 s
Press any key to continue.