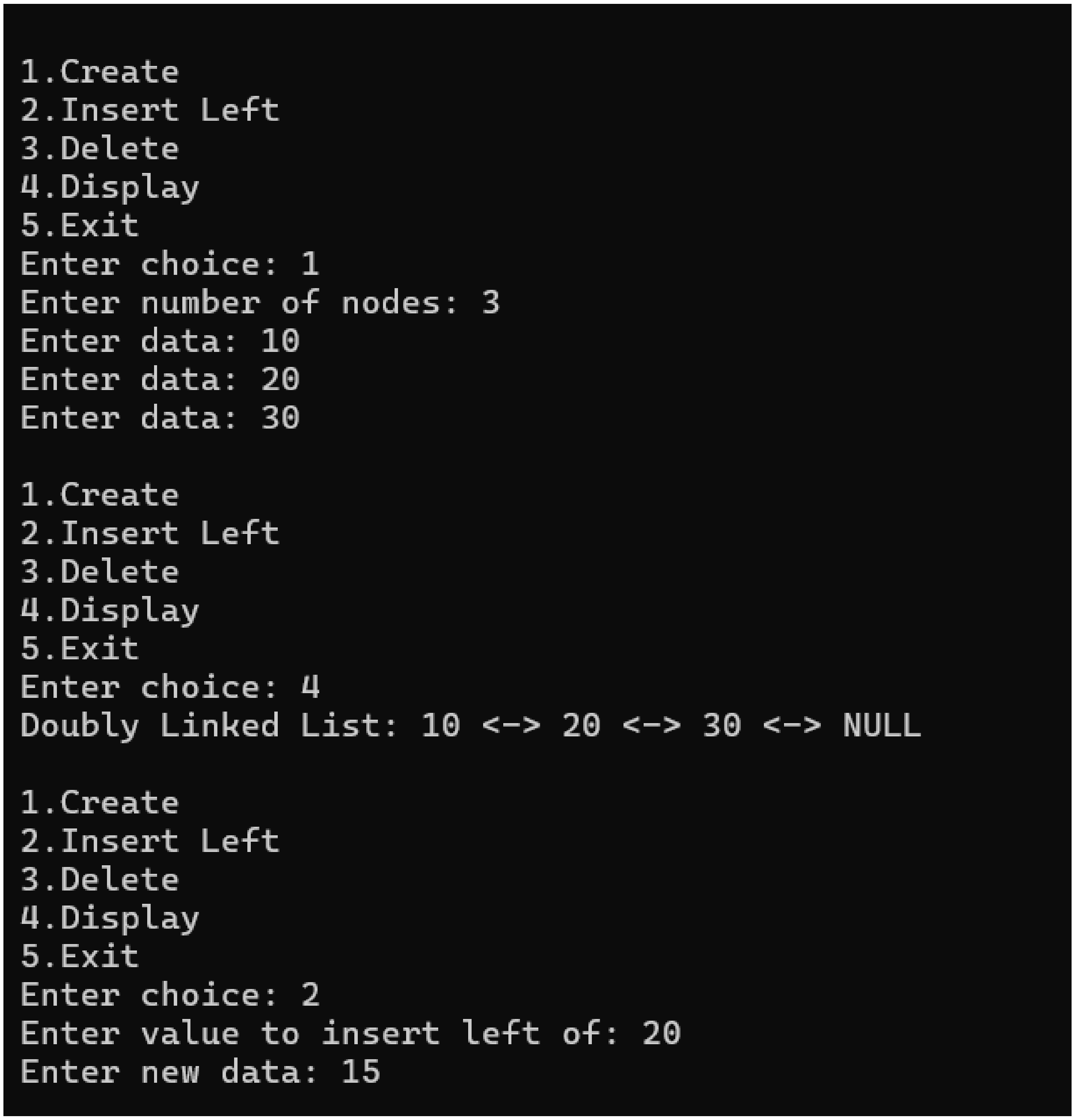
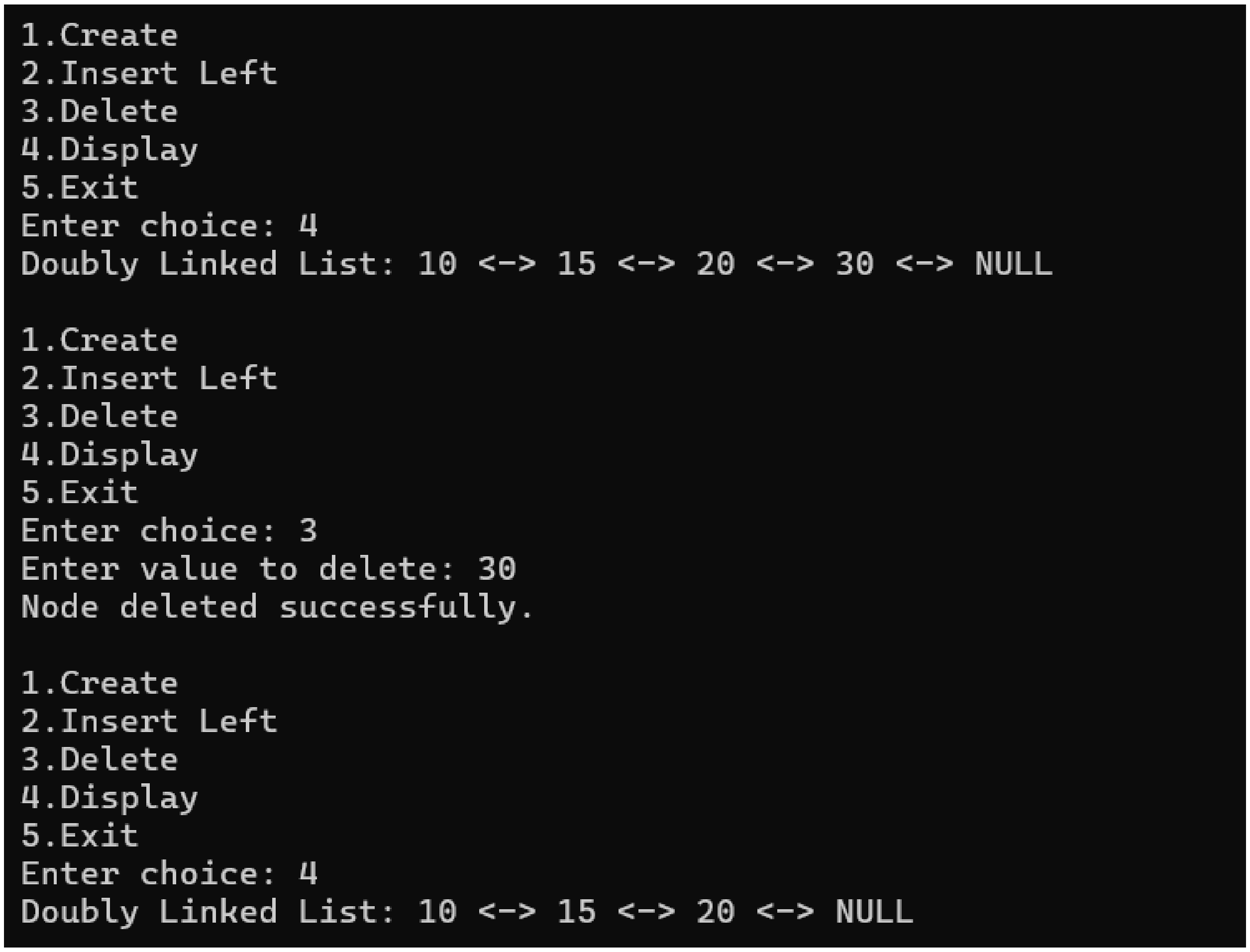
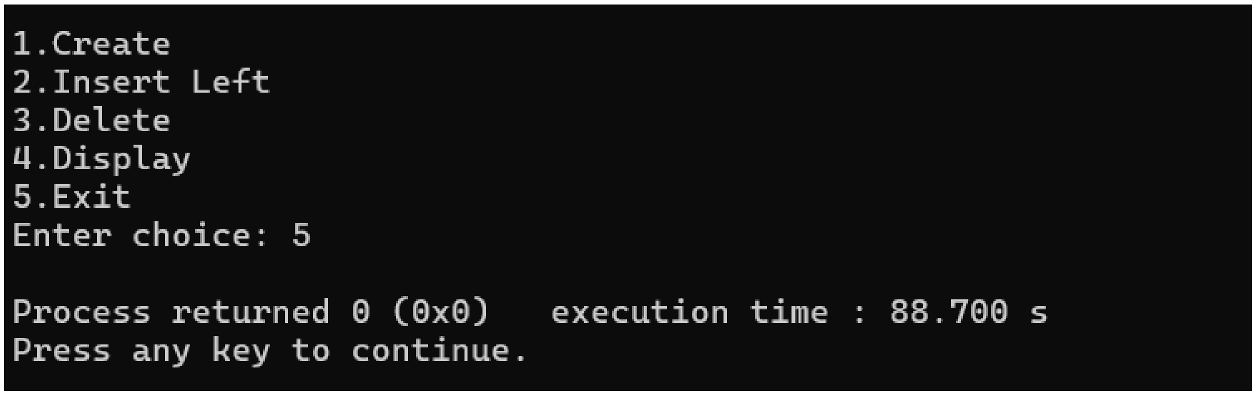
7(a)-WAP to Implement doubly link list with primitive operations a) Create a doubly linked list. b) Insert a new node to the left of the node. c) Delete the node based on a specific value d) Display the contents of the list

```
#include <stdio.h>

#include <stdlib.h>


// Structure definition

struct node {

    int data;

    struct node *prev;

    struct node *next;

};


struct node *head = NULL;


// Create doubly linked list

void create() {

    struct node *newnode, *temp;

    int n, i;


    printf("Enter number of nodes: ");

    scanf("%d", &n);


    for (i = 0; i < n; i++) {

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

        printf("Enter data: ");

        scanf("%d", &newnode->data);
        newnode->prev = NULL;

        newnode->next = NULL;


        if (head == NULL) {

            head = newnode;

            temp = head;

        } else {

            temp->next = newnode;

            newnode->prev = temp;

            temp = newnode;

        }

    }

}


// Insert a node to the left of a given value

void insert_left() {

    struct node *newnode, *temp;

    int value;


    printf("Enter value to insert left of: ");

    scanf("%d", &value);


    temp = head;

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

        temp = temp->next;

    }


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

    newnode = (struct node *)malloc(sizeof(struct node));
    printf("Enter new data: ");
    scanf("%d", &newnode->data);

    newnode->next = temp;
    newnode->prev = temp->prev;

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

    temp->prev = newnode;
}

// Delete a node with specific value
void delete_node() {
    struct node *temp;
    int value;

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

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

    if (temp == NULL) {
        printf("Value not found!\n");
        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 deleted successfully.\n");
}

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

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

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

// Main function
int main() {
    int choice;

    while (1) {
        printf("\n1.Create\n2.Insert Left\n3.Delete\n4.Display\n5.Exit\n");
        printf("Enter 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:
            exit(0);
        default:
            printf("Invalid choice!\n");
        }
    }
    return 0;
}
```


```
1.Create
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 1
Enter number of nodes: 3
Enter data: 10
Enter data: 20
Enter data: 30

1.Create
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 4
Doubly Linked List: 10 <-> 20 <-> 30 <-> NULL

1.Create
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 2
Enter value to insert left of: 20
Enter new data: 15
```



```
1.Create
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 4
Doubly Linked List: 10 <-> 15 <-> 20 <-> 30 <-> NULL

1.Create
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 3
Enter value to delete: 30
Node deleted successfully.

1.Create
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 4
Doubly Linked List: 10 <-> 15 <-> 20 <-> NULL
```



```
1.Create
2.Insert Left
3.Delete
4.Display
5.Exit
Enter choice: 5

Process returned 0 (0x0)   execution time : 88.700 s
Press any key to continue.
```