LAB-7 (WAP to Implement doubly link list with primitive operation
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
Display the contents of the list)

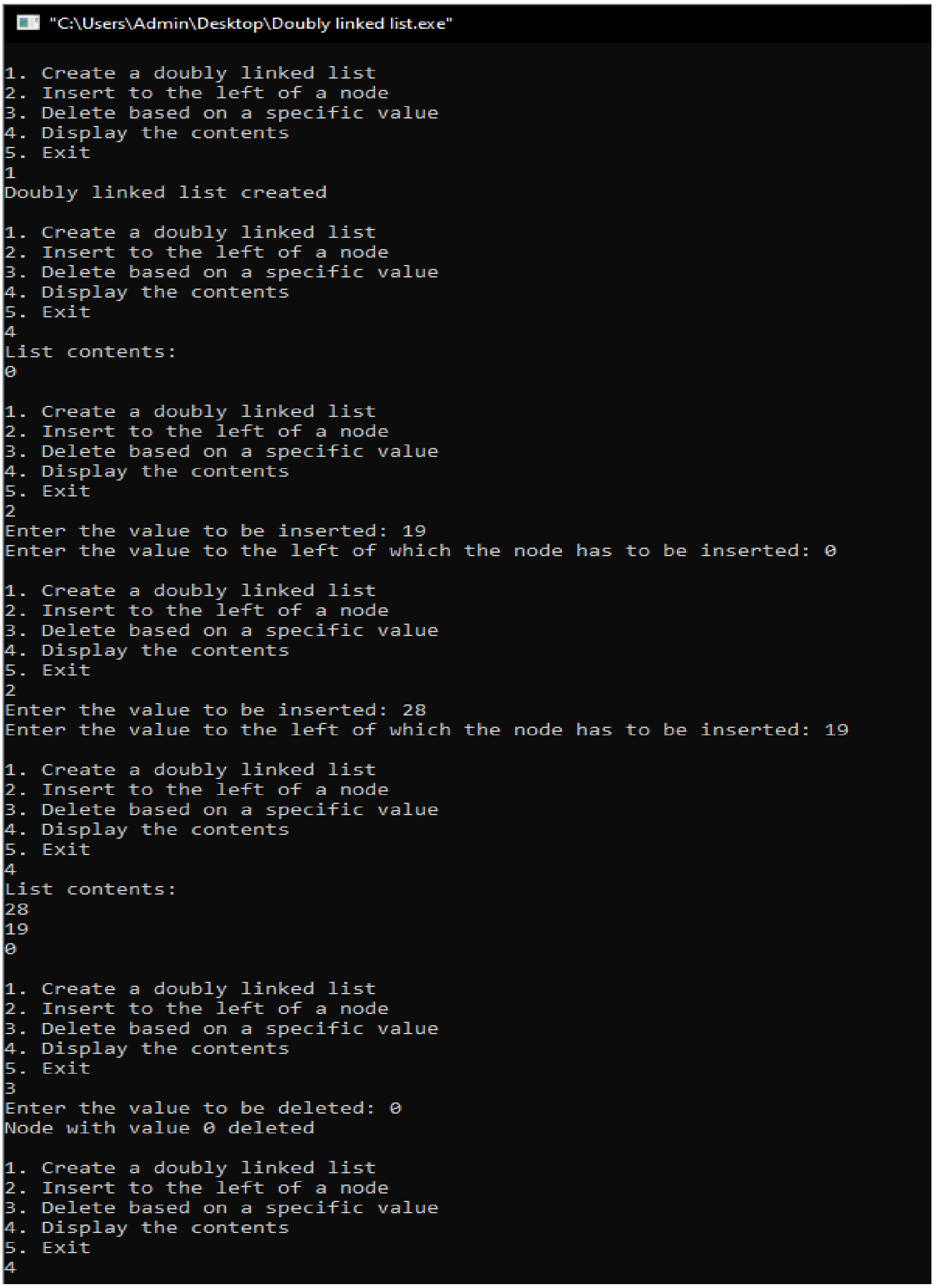
SOURCE CODE:

```
#include<stdio.h>
#include<stdlib.h>
struct node {
   int data;
   struct node *prev;
   struct node *next;
};
struct node *s1 = NULL;
struct node *createNode(int value) {
   struct node *temp = (struct node *)malloc(sizeof(struct node));
   temp->data = value;
   temp->next = NULL;
   temp->prev = NULL;
   return temp;
}
struct node *insert_left(struct node *start) {
   int value, key;
   struct node *temp = createNode(0);
   printf("Enter the value to be inserted: ");
   scanf("%d", &temp->data);
   printf("Enter the value to the left of which the node has to be inserted: ");
   scanf("%d", &key);
   struct node *ptr = start;
   while (ptr != NULL && ptr->data != key) {
      ptr = ptr->next;
   }
   if (ptr == NULL) {
      printf("Node with value %d not found\n", key);
      free(temp);
   } else {
      temp->next = ptr;
      temp->prev = ptr->prev;
      if (ptr->prev != NULL) {
         ptr->prev->next = temp;
      }
      ptr->prev = temp;
      if (ptr == start) {
         start = temp;
      }
   }
   return start;
}
struct node *delete_value(struct node *start) {
   int value;
   printf("Enter the value to be deleted: ");
   scanf("%d", &value);
```

```
    struct node *ptr = start;
    while (ptr != NULL && ptr->data != value) {
        ptr = ptr->next;
    }
    if (ptr == NULL) {
        printf("Node with value %d not found\n", value);
    } else {
        if (ptr->prev != NULL) {
            ptr->prev->next = ptr->next;
        } else {
            start = ptr->next;
        }
        if (ptr->next != NULL) {
            ptr->next->prev = ptr->prev;
        }
        printf("Node with value %d deleted\n", value);
        free(ptr);
    }
    return start;
}
void display(struct node *start) {
    struct node *ptr = start;
    if (start == NULL) {
        printf("List is empty\n");
    } else {
        printf("List contents:\n");
        while (ptr != NULL) {
            printf("%d\n", ptr->data);
            ptr = ptr->next;
        }
    }
}
int main() {
    int choice;
    while (1) {
        printf("\n1. Create a doubly linked list\n2. Insert to the left of a node\n3. Delete based on a
specific value\n4. Display the contents\n5. Exit\n");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                s1 = createNode(0);
                printf("Doubly linked list created\n");
                break;
            case 2:
                s1 = insert_left(s1);
                break;
            case 3:
                s1 = delete_value(s1);
                break;
            case 4:
                display(s1);
                break;
            case 5:
```

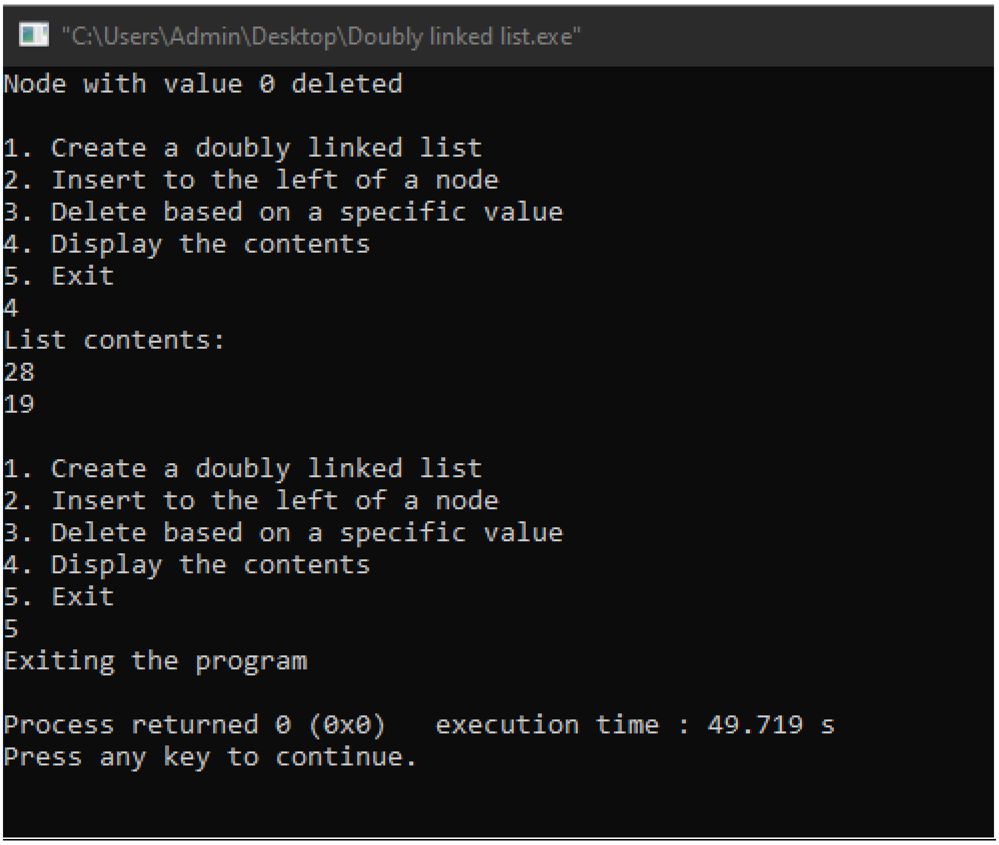
```
            printf("Exiting the program\n");
            exit(0);
          default:
            printf("Invalid choice\n");
        }
    }
    return 0;
}
```

OUTPUT:



"C:\Users\Admin\Desktop\Doubly linked list.exe"

```
1. Create a doubly linked list
2. Insert to the left of a node
3. Delete based on a specific value
4. Display the contents
5. Exit
1
Doubly linked list created

1. Create a doubly linked list
2. Insert to the left of a node
3. Delete based on a specific value
4. Display the contents
5. Exit
4
List contents:
0

1. Create a doubly linked list
2. Insert to the left of a node
3. Delete based on a specific value
4. Display the contents
5. Exit
2
Enter the value to be inserted: 19
Enter the value to the left of which the node has to be inserted: 0

1. Create a doubly linked list
2. Insert to the left of a node
3. Delete based on a specific value
4. Display the contents
5. Exit
2
Enter the value to be inserted: 28
Enter the value to the left of which the node has to be inserted: 19

1. Create a doubly linked list
2. Insert to the left of a node
3. Delete based on a specific value
4. Display the contents
5. Exit
4
List contents:
28
19
0

1. Create a doubly linked list
2. Insert to the left of a node
3. Delete based on a specific value
4. Display the contents
5. Exit
3
Enter the value to be deleted: 0
Node with value 0 deleted

1. Create a doubly linked list
2. Insert to the left of a node
3. Delete based on a specific value
4. Display the contents
5. Exit
4
```



```
Node with value 0 deleted

1. Create a doubly linked list
2. Insert to the left of a node
3. Delete based on a specific value
4. Display the contents
5. Exit
4
List contents:
28
19

1. Create a doubly linked list
2. Insert to the left of a node
3. Delete based on a specific value
4. Display the contents
5. Exit
5
Exiting the program

Process returned 0 (0x0)   execution time : 49.719 s
Press any key to continue.
```