<DOUBLY LINKED LIST>

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

ANSWER: -

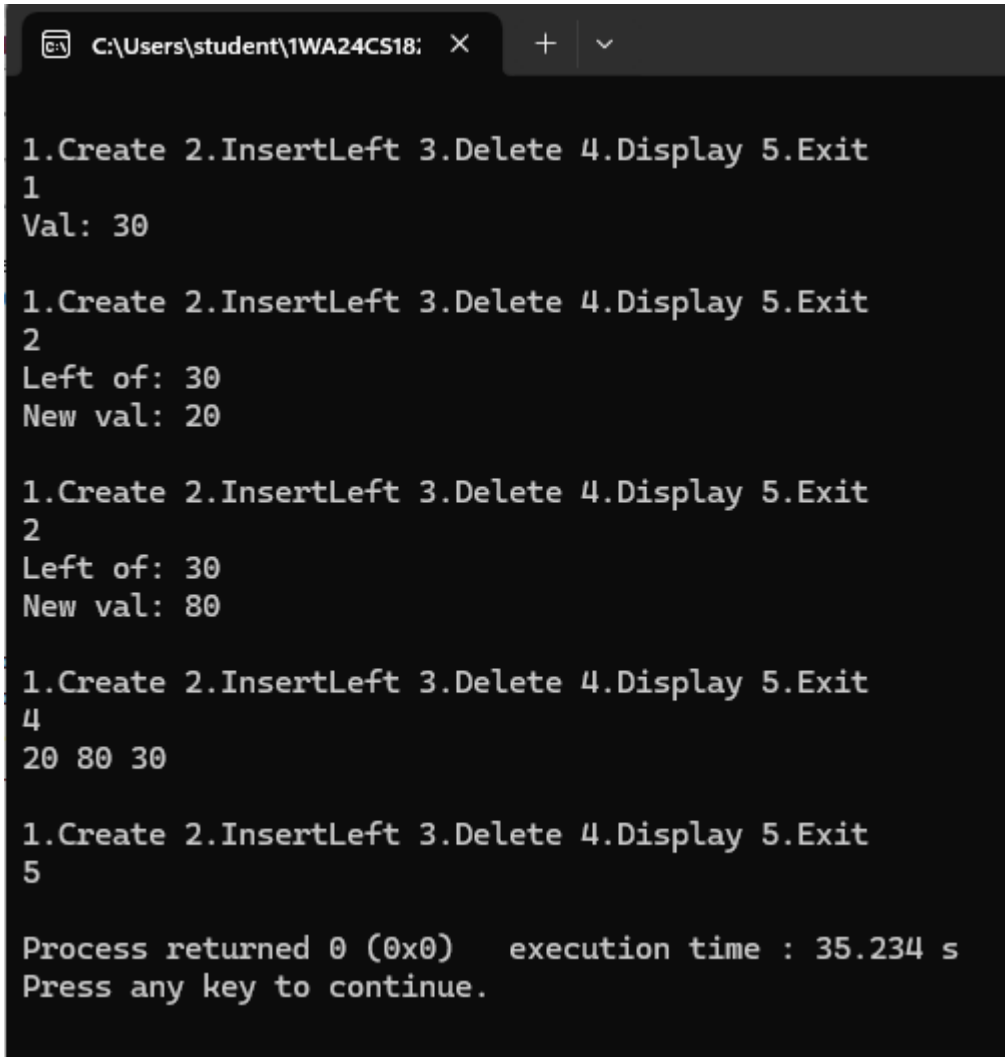
```c
#include <stdio.h>
#include <stdlib.h>

typedef struct node{
    int data;
    struct node *prev,*next;
}Node;

Node *head=NULL;

Node* newNode(int x){
    Node *n = (Node*)malloc(sizeof(Node));
    if(!n){
        printf("Memory allocation failed\n");
        exit(1);
    }
    n->data = x;
    n->prev = n->next = NULL;
    return n;
}

void create(int x){
    Node *n = newNode(x), *t = head;

    if(!head){
        head = n;
    }
    else{
        while(t->next)
            t = t->next;

        t->next = n;
        n->prev = t;
    }
}
```

```
37  void insertLeft(int val,int x){
38      Node *t = head;
39      while(t && t->data != val)
40          t = t->next;
41
42      if(!t){
43          printf("Not found\n");
44          return;
45      }
46
47      Node *n = newNode(x);
48      n->next = t;
49      n->prev = t->prev;
50
51      if(t->prev)
52          t->prev->next = n;
53      else
54          head = n;
55
56      t->prev = n;
57  }
58
59  void deleteVal(int val){
60      Node *t = head;
61      while(t && t->data != val)
62          t = t->next;
63
64      if(!t){
65          printf("Not found\n");
66          return;
67      }
68
69      if(t->prev)
70          t->prev->next = t->next;
71      else
72          head = t->next;
73
```

```
73
74          if(t->next)
75              t->next->prev = t->prev;
76
77          free(t);
78      }
79
80  void display(){
81          Node *t = head;
82          if(!t){
83              printf("Empty\n");
84              return;
85          }
86
87          while(t){
88              printf("%d ", t->data);
89              t = t->next;
90          }
91          printf("\n");
92      }
```

```
94  int main(){
95      int ch,val,x;
96
97      while(1){
98          printf("\n1.Create 2.InsertLeft 3.Delete 4.Display 5.Exit\n");
99          scanf("%d",&ch);
100
101         if(ch==1){
102             printf("Val: ");
103             scanf("%d",&x);
104             create(x);
105         }
106         else if(ch==2){
107             printf("Left of: ");
108             scanf("%d",&val);
109             printf("New val: ");
110             scanf("%d",&x);
111             insertLeft(val,x);
112         }
113         else if(ch==3){
114             printf("Delete val: ");
115             scanf("%d",&val);
116             deleteVal(val);
117         }
118         else if(ch==4){
119             display();
120         }
121         else if(ch==5){
122             break;
123         }
124         else{
125             printf("Invalid\n");
126         }
127     }
128
129     return 0;
130 }
```

OUTPUT: -


C:\Users\student\1WA24CS18:
1.Create 2.InsertLeft 3.Delete 4.Display 5.Exit
1
Val: 30
1.Create 2.InsertLeft 3.Delete 4.Display 5.Exit
2
Left of: 30
New val: 20
1.Create 2.InsertLeft 3.Delete 4.Display 5.Exit
2
Left of: 30
New val: 80
1.Create 2.InsertLeft 3.Delete 4.Display 5.Exit
4
20 80 30
1.Create 2.InsertLeft 3.Delete 4.Display 5.Exit
5
Process returned 0 (0x0) execution time : 35.234 s
Press any key to continue.