PROGRAM 2
IMPLEMENTATION OF DOUBLY LINKED LIST

ALGORITHM:

Step 1: Start

Step 2: Declare structure
```
typedef struct node {
        int data;
        struct node *prev;
        struct node *next;
}listNode;
```

Step 3: listnode *beg = NULL;

Step 4: listnode *end = NULL;

Step 5: Print Menu

Step6: Declare choice (int) and input values from user

Step7: Check if(choice!=6)
if true proceed to step 8 else go to step 9

Step 8: switch(choice){
case 1: call insertElement(1);break;
case 2: call insertElement(-1);break;
case 3: call insertElement(0);break;
case 4: call delete_ele(); break;
case 5: call display(); break;
case 6: exit
default : Print “Enter a valid choice”
}

Step 9: Stop

InsertElement( ):
Algo

InsertElement(int pos)
Step 1. input ‘data’ to be inserted from the user.
Step 2. Allocate memory for a new Node* variable ‘n’ with data = data
Declare prev and next to NULL
Step 3. If (pos == 0), input the position from the user; if pos <= 0, make pos = 1;

Step 4. If (‘beg’ == NULL and ‘end’ == NULL), make beg = end = n;
Step 5. Else If (pos == 1)execute :
5.1 beg->prev = n;
5.2 n->next = beg;
5.3 beg = n;
Step 6: Else,
6.1. Initialize struct Node* variable ‘temp’ as ‘beg’.
6.2. Initialize an int variable ‘count’ = 0.
6.3. While(temp != NULL)
6.3.1 count++;
If (count == pos),
break; temp = temp->next;
6.4. If (temp!=NULL),
n->prev = temp->prev;
temp->prev->next = n;
n->next = temp;
temp->prev = n;
6.5 Else,
end->next = n;
n->prev = end;
end = n;
Step 7: Execute displayList().
Step 8: Stop

DeleteElement() :
Algo

Step 1: If (‘beg’ == NULL and ‘end’ == NULL)
Print “underflow”
Step 2. Input the element to delete from the user as ‘ele’.
Step 3. Initialize a struct Node* variable ‘temp’ with ‘beg’.
Step 4. While (temp != NULL) Do
4.1. If(temp->data = ele)
4.1.1 If(temp == beg and temp == end)
Set beg = end = NULL.
4.1.2 Else If (temp == beg)
beg = beg->next;
beg->prev = NULL;
4.1.3 Else If (tem == end)
end = end->prev;
end->next = NULL;
4.1.4 Else
temp->prev->next = temp->next;
temp->next->prev = temp->prev;
free ‘temp’ from memory and break.
4.2. Set temp = temp->next.
Step 5: Set temp = temp->next
Step 6: Stop