

1. WAP to implement doubly link list with primitive operations.

- Create a doubly linked list.
- Insert a new node to the left of the node.
- Delete the node based on a specific value.

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

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

```
struct node
```

```
{
```

```
    int data;
```

```
    struct node *next;
```

```
    struct node *prev;
```

```
};
```

```
struct node *head;
```

```
void create_ll()
```

```
{
```

```
    struct node *new_node, *ptr;
```

```
    int num;
```

```
    printf("Enter -1 to exit. \n");
```

```
while (num != -1)
```

```
{
```

```
    printf ("Enter the num: ");
```

```
    scanf ("%d", &num);
```

```
    new_node = (struct node *) malloc (sizeof (struct node));
```

```
    new_node -> data = num;
```

```
    if (head == NULL)
```

```
    {
```

```
        head = new_node;
```

```
        new_node -> next = NULL;
```

```
        new_node -> prev = NULL;
```

```
    }
```

```
edge
```

```
{
```

```
    ptr = head;
```

```
    while (ptr -> next != NULL)
```

```
    {
```

```
        ptr = ptr -> next;
```

```
    }
```

```
    ptr -> next = new_node;
```

```
    new_node -> prev = ptr;
```

```
    new_node -> next = NULL;
```

```
    }
```

```
}
```

```
}
```

void insertatJedf()

{

struct node *newnode, *ptr;

int val, num;

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

printf ("Enter a value to insert at Jedf: \n");

scanf ("%d", &val);

printf ("Enter the val of node: ");

scanf ("%d", &num);

newnode->data = val;

ptr = head;

while (ptr->data != num)

{

ptr = ptr->next;

}

ptr->prev->next = newnode;

newnode->prev = ptr->prev;

newnode->next = ptr;

ptr->prev = newnode;

}


```
void display()
```

```
{
```

```
    struct node *ptr;
```

```
    if (head == NULL)
```

```
    {
```

```
        printf("LL is empty");
```

```
    }
```

```
    else
```

```
    {
```

```
        ptr = head;
```

```
        while (ptr->next != NULL)
```

```
        {
```

```
            printf("%d", ptr->data);
```

```
            ptr = ptr->next;
```

```
        }
```

```
    }
```

```
}
```

```
void del()
```

```
{
```

```
    struct node *ptr;
```

```
    int value;
```

```
    printf("Enter the value to be deleted: ");
```

```
    scanf("%d", &val);
```

```
    ptr = head;
```

```
if (head->data == val)
```

```
{
```

```
    ptr = ptr->next;
```

```
    head = ptr;
```

```
}
```

```
else
```

```
{
```

```
    while (ptr->data != val)
```

```
    {
```

```
        ptr = ptr->next;
```

```
    }
```

```
    ptr->prev->next = ptr->next;
```

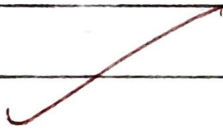
```
    ptr->next->prev = ptr->prev;
```

```
}
```

```
force(ptr);
```

```
}
```

```
}
```



Output:

- - - - Menu - - - -

1. create LL
2. Insert - left
3. delete
4. display
5. exit

1

Enter -1 to exit

Enter the num: 1

Enter the num: 2

Enter the num: 3

Enter the num: 4

Enter the num: -1

4

1 -> 2 -> 3 -> 4

For
5/2/24

2

enter the val to insert d left: 99

enter the val of node: 46

4

~~1~~ -> 99 -> 2 -> 3 -> 4

3

Enter the value to be deleted: 99 4 | -1 -> 2 -> 3 -> 4