

Aim:

Write a C program that uses functions to perform the following **operations on double linked list**

i) Creation ii) Insertion iii) Deletion iv) Traversal

Source Code:**AllOperationsDLL.c**

```
#include<stdio.h>
#include<stdlib.h>
void insert();
void rem();
void display();
struct node
{
    int data;
    struct node *next;
    struct node*prev;
}
*head = NULL, *tail = NULL;
typedef struct node *NODE;
void main()
{
    int option = 0;
    while(1)
    {
        printf("Operations on doubly linked list\n");
        printf("1. Insert \n");
        printf("2.Remove\n");
        printf("3. Display\n");
        printf("0. Exit\n");
        printf("Enter Choice 0-4? : ");
        scanf("%d",&option);
        switch(option)
        {
            case 1:
                insert();
                break;
            case 2:
                rem();
                break;
            case 3:
                display();
                break;
            case 0:
                exit(0);
        }
    }
}

void insert()
{
    NODE temp, newNode;
    int value;
```

```
newNode = (NODE)malloc(sizeof(struct node));
printf("Enter number: ");
scanf("%d",&value);
newNode->data = value;
if(head == NULL)
{
    newNode->next = NULL;
    newNode->prev = NULL;
    head = newNode;
    tail = newNode;
}
else
{
    tail->next = newNode;
    newNode->prev = tail;
    newNode->next = NULL;
    tail = newNode;
}
}
void rem()
{
    int devalue,item;
    NODE temp,ptr;
    printf("Enter number to delete: ");
    scanf("%d",&item);
    ptr = head;
    while(ptr != NULL)
    {
        if(ptr->data == item)
        {
            devalue = item;
            break;
        }
        ptr = ptr->next;
    }
    if(devalue != item)
    printf("%d not found.\n",item);
    else
    {
        if(devalue == head->data)
        {
            temp=head;
            head=head->next;
            head->prev=NULL;
            free(temp);
        }
        else
        {
            temp=head;
            while(temp->data != devalue)
            {
                temp=temp->next;
            }
            temp->prev->next=temp->next;
            temp->next->prev=temp->prev;
            free(temp);
        }
    }
}
```

```

    }
}
void display()
{
    NODE temp;
    temp = head;
    while(temp !=NULL)
    {
        printf("%d\t",temp->data);
        temp=temp->next;
    }
    printf("\n");
}

```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Operations on doubly linked list 1
1.Insert 1
2.Remove 1
3.Display 1
0.Exit 1
Enter Choice 0-4?: 1
Enter number: 15
Operations on doubly linked list 1
1.Insert 1
2.Remove 1
3.Display 1
0.Exit 1
Enter Choice 0-4?: 1
Enter number: 16
Operations on doubly linked list 1
1.Insert 1
2.Remove 1
3.Display 1
0.Exit 1
Enter Choice 0-4?: 1
Enter number: 17
Operations on doubly linked list 1
1.Insert 1
2.Remove 1
3.Display 1
0.Exit 1
Enter Choice 0-4?: 1
Enter number: 18
Operations on doubly linked list 3
1.Insert 3
2.Remove 3
3.Display 3
0.Exit 3

Enter Choice 0-4?: 3
15 16 17 18 2
Operations on doubly linked list 2
1.Insert 2
2.Remove 2
3.Display 2
0.Exit 2
Enter Choice 0-4?: 2
Enter number to delete: 19
19 not found 3
Operations on doubly linked list 3
1.Insert 3
2.Remove 3
3.Display 3
0.Exit 3
Enter Choice 0-4?: 3
15 16 17 18 2
Operations on doubly linked list 2
1.Insert 2
2.Remove 2
3.Display 2
0.Exit 2
Enter Choice 0-4?: 2
Enter number to delete: 16
Operations on doubly linked list 0
1.Insert 0
2.Remove 0
3.Display 0
0.Exit 0
Enter Choice 0-4?: 0