S.No: 14	Exp. Name: ***C program which performs all operations on double linked list.***	Date:2023-07-01

ID: 224G1A0543

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>
#include <conio.h>

struct dnode
{
       struct dnode *prev;
       int data;
       struct dnode *next;
};

struct dnode *start = NULL;

void insert(int);
void remov(int);
void display();

int main()
{
       int n, ch;
        do
        {
             printf("Operations on doubly linked list");
             printf("\n1. Insert \n2.Remove\n3. Display\n0. Exit");
             printf("\nEnter Choice 0-4? : ");
             scanf("%d", &ch);
             switch (ch)
             {
                 case 1:
                      printf("Enter number: ");
                      scanf("%d", &n);
                      insert(n);
                      break;
                   case 2:
                      printf("Enter number to delete: ");
                      scanf("%d", &n);
                      remov(n);
                      break;
                   case 3:
                       display();
                        break;
             }
        }while (ch != 0);
}
void insert(int num)
```

```
{
       struct dnode *nptr,  *temp = start;


        nptr = malloc(sizeof(struct dnode));
        nptr->data = num;
        nptr->next = NULL;
        nptr->prev = NULL;


         if (start == NULL)
         {
             start = nptr;
          }
    else
        {
          while (temp->next != NULL)
              temp = temp->next;

              nptr->prev = temp;
           temp->next = nptr;
        }
}

void remov(int num)
{
    struct dnode *temp = start;

   while (temp != NULL)
   {
      if (temp->data == num)
       {
           if (temp == start)
           {
               start = start->next;
               start->prev = NULL;
           }
           else
           {
                if (temp->next == NULL)
                  temp->prev->next = NULL;
               else
               {
                  temp->prev->next = temp->next;
                    temp->next->prev = temp->prev;
                 }
                 free(temp);
               }
              return ;
           }
           temp = temp->next;
       }
       printf("%d not found.\n", num);
}
void display()
{
```

```
    struct dnode *temp = start;
    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

ID: 224G1A0543

```
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
```