EXPERIMENT 2

2.1 OBJECTIVE

1. Create a doubly linked list of integers.
2. Delete a given integer from the above doubly linked list.
3. Display the contents of the above list after deletion.

2.2 RESOURCE

Turbo C

2.3 PROGRAM LOGIC

1. Create a node using structure
2. Dynamically allocate memory to node
3. Create and add nodes to linked list

2.4 PROCEDURE

Go to debug -> run or press CTRL + F9 to run the program

2.5 SOURCE CODE

Program to create a double linked list to inserting, deleting and displaying the contents

```
#include<stdio.h>
#include<stdlib.h>
/*declaring a structure to create a node*/
struct node
{
     struct node *prev;
     int data;
     struct node *next;
};
struct node *start,*nt;
/* inserting nodes into the list*/
/*function to insert values from beginning of the the double linked list*/

void insertbeg(void)
{
    int a;
    struct node *nn,*temp;
/*allocating implicit memory to the node*/
    nn=(struct node *)malloc(sizeof(struct node));
    printf("enter data:");
    scanf("%d",&nn->data);
    a=nn->data;
   if(start==NULL)     /*checking if List is empty*/
  {
     nn->prev=nn->next=NULL;
     start=nn;
  }
  else
 {
    nn->next=start;
    nn->prev=NULL;
    start->prev=nn;
    start=nn;
  }
  printf("%d succ inserted \n",a);
```

```
}
/*function to insert values from the end of the linked list*/

void insertend(void)
{
    int b;
    struct node *nn,*lp;
    nn=(struct node *)malloc(sizeof(struct node));
    printf("enter data:");
    scanf("%d",&nn->data);
    b=nn->data;
    if(start==NULL)
   {
/* assigning first node pointer to next nod pointer to delete a data from the starting
   of the node*/

       nn->prev=nn->next=NULL;
       start=nn;
   }
   else
  {
     lp=start;
    while(lp->next!=NULL)
    {
       lp=lp->next;
     }
    nn->prev=lp;
    lp->next=nn;
    nn->next=NULL;
  }
 printf("%d succ inserted\n",b);
}
/*function to insert values from the middle of the linked list*/

void insertmid(void)
{
    struct node *nn,*temp,*ptemp;
    int x,c;
   if(start==NULL)
   {
       printf("dll is empty\n");
      return;
   }
    printf("enter data before which nn is to be inserted\n");
    scanf("%d",&x);
    if(x==start->data)
    {
       insertbeg();
     }
     ptemp=start;
     temp=start->next;
     while(temp->next!=NULL&&temp->data!=x)
    {
      ptemp=temp;
      temp=temp->next;
    }
    if(temp==NULL)
   {
```

```
        printf("%d does not exit\n",x);
      }
      else
     {
/*allocating implicit memory to the node*/

         nn=(struct node *)malloc(sizeof(struct node));
         printf("enter data");
         scanf("%d",&nn->data);
        c=nn->data;
         nn->data;
       nn->prev=ptemp;
       nn->next=temp;
       ptemp->next=nn;
       temp->prev=nn;
       printf("%d succ inserted \n",c);
   }
}
/*end of insertion operation*/
/*deletion operation*/
void deletion()
{
     struct node *pt,*t;
     int x;
     t=pt=start;
     if(start==NULL)
    {
        printf("dll is empty\n");
     }
     printf("enter data to be deleted:");
    scanf("%d",&x);
    if(x==start->data)
   {
      t=start;
      t=t->next;
      free(start);
      start=t;
      start=pt;
   }
   else
  {
     while(t->next!=NULL&&t->data!=x)
    {
       pt=t;    /*logic for traversing*/
       t=t->next;
    }
    if(t->next==NULL&&t->data==x)
   {
      free(t);
     pt->next=NULL;
   }
   else
   {
     if(t->next==NULL&&t->data!=x)
         printf("data not found");
    else
     {
        pt->next=t->next;
```

```
        free(t);
      }
    }
   printf("%d is succ deleted\n",x);
  }
}
/*end of deletion operation*/
/*display operation*/
void display()
{
    struct node *temp;
    if(start==NULL)
    printf("stack is empty ");
    temp=start;
    while(temp->next!=NULL)
   {
      printf("%d",temp->data);
      temp=temp->next;
   }
   printf("%d",temp->data);
 }
/*end of display operation*/
/*main program*/
int main()
{
    int c,a;
    start=NULL;
   do
  {
      printf("1.insert\n2.delete\n3.display\n4.exit\nenter choice:");
      scanf("%d",&c);
      switch(c)
      {
         case 1:printf("1.insertbeg\n2.insertend\n3.insertmid\nenter choice:");
                scanf("%d",&a);
                switch(a)
                {
                      case 1:insertbeg();
                             break;
                      case 2:insertend();
                             break;
                      case 3:insertmid();
                             break;
                }
               break;
        case 2:deletion();
                break;
       case 3:display();
                break;
      case 4:printf("program ends\n");
               break;
      default:printf("wrong choice\n");
               break;
    }
 }
  while(c!=4);
  return 0;
 }
```

2.6 PRE LAB QUESTIONS

1. What is double linked list
2. How to represent a node in double linked list
3. Differentiate between single and double linked list

2.7 LAB ASSIGNMENT

1. Write a program to insert a node at first , last and at specified position of double linked list
2. Write a program to eliminate duplicates from double linked list
3. Write a program to delete a node from first, last and at specified position of double linked list

2.8 POST LAB QUESTIONS

1. How to represent double linked list
2. How will you traverse double linked list
3. List the advantages of double linked list over single list

2.9 INPUT AND OUTPUT