```
/**********************SUBHAS NATH*******************/
/****************************************************/
/************PROGRAM OF DOUBLE LINKED LIST**********/
/*********INSERTION & DELETION  (ANY POSITION)********/

#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#define NULL 0
struct link
 {
  struct link *prev;
  int info;
  struct link *next;
 };
typedef struct link node;

void create(node *);
void show(node *);
node* insert(node*);//INSERTING IN DLL
node* del(node*);//DELETION IN DLL
node* find (node *,int);//FINDING NODE

void main()
 {
  node *head;
  clrscr();
  head=(node *)malloc(sizeof(node)); //CREATE STORAGE FOR FIRST NODE

  printf("\nCreate the linked list:\n");
  create(head);

  printf("\nDisplay the linked list:\n");
  show(head);

  getch();
  clrscr();

  printf("\nInsert number at any position in the given linked list:\n");
  show(head);
  head=insert(head);
  show(head);

  getch();
  clrscr();
```

```
 printf("\nDelete number from any position in the given linked list:\n");
 show(head);
 head=del(head);
 show(head);
 getch();
}

/*******************CREATE LINKED LIST*******************/

void create(node *temp)
 {
 char ans;
 node *p;
 temp->prev=NULL;
 printf("\n\tEnter the info:-   ");
 scanf("%d",&temp->info);
 while(1)
  {
  printf("\n\tWant Another?  ");
  fflush(stdin); //CLEAR THE BUFFER
  ans=getchar();
  if(ans!='y')
   break;
  else
   {
    p=temp;
    temp->next=(node *)malloc(sizeof(node));
    temp=temp->next;
    temp->prev=p;
    printf("\n\tEnter the info:-   ");
    scanf("%d",&temp->info);
   }
  }
 temp->next=NULL;
}

/*******************DISPLAY LINKED LIST*******************/

void show(node *temp)
 {
 printf("\n\t");
 while(temp->next!=NULL)
  {
  printf("%d<->",temp->info); //DISPLAY UPTO PREVIOUS OF LAST NODE
  temp=temp->next;
  }
```

```
 if(temp->next==NULL)
  {
   printf("%d",temp->info);//DISPLAY LAST NODE
  }
 printf("\n\n\t");
 while(temp->prev!=NULL)
  {
   printf("%d<->",temp->info); //DISPLAY FROM LAST TO SECOND NODE
   temp=temp->prev;
  }
 if(temp->prev==NULL)
  {
   printf("%d",temp->info);//DISPLAY FIRST NODE
  }
}

/******************INSERT AT ANY POSITION*****************/

node *insert(node *temp)
 {
 int key;
 node *mid,*n1,*first;
 first=temp;
 mid=(node *)malloc(sizeof(node));
 printf("\n\nInsert Before which number?");
 printf("\nNote:: (Give -9999 if INSERTING at the END of list!)");
 scanf("%d",&key);
 printf("\nNumber to be Inserted:");scanf("%d",&mid->info);


 //IF THE KEY AT THE 1ST POSITION.
 if(temp->info==key)
  {
   mid->prev=NULL;
   mid->next=temp;
   temp->prev=mid;
   temp=mid;
  }
 else
  {
   n1=find(temp,key);

   //IF KEY IS NOT PRESENT IN THE LIST.
   if(n1==NULL)
    {
     printf("\n\tKey not FOUND, Inserting at the END!\n");
```

```
      while(temp->next!=NULL)
       temp=temp->next;

      mid->prev=temp;
      mid->next=NULL;
      temp->next=mid;
      temp=first;
     }
    //INSERT  ON THE MIDDLE OF THE LIST.
    else
     {
      mid->prev=n1;
      mid->next=n1->next;
      n1->next=mid;
      mid->next->prev=mid;
     }
   }
  return(temp);
 }


/***************SEARCHING THE KEY IN THE LIST**************/

node *find(node *temp,int key)
 {
  if(temp->next->info==key)
     return(temp);
  else if(temp->next->next==NULL)
     return(NULL);
  else
     return(find(temp->next,key));
 }


/*****************DELETION FROM ANY POSITION*************/

node *del(node *temp)
 {
  int key;
  node *first,*p,*q;
  first=temp;
  printf("\n\nDelete which number?  ");
  scanf("%d",&key);
```

```
 while(temp->next!=NULL)
   {
   if(temp->info==key)
    {
    //DELETE FROM MIDDLE POSITION.
     if(temp->prev!=NULL)
      {
          p=temp->prev;
          q=temp->next;
          free(temp);
          p->next=q;
          q->prev=p;
      }
    //DELETE FROM FIRST POSITION.
     else
      {
          temp=temp->next;
          temp->prev=NULL;
          first=temp;
      }
    }
   temp=temp->next;
   }

//DELETE FROM LAST POSITION.
  if((temp->next==NULL)&&(temp->info==key))
   {
   temp=temp->prev;
   temp->next=NULL;
   }

 temp=first;
 return(temp);
}
```