```
#include<stdio.h>
#include<conio.h>
#include<alloc.h>
#include<stdlib.h>
#define NULL 0

struct linkedlist
{
int item;
struct linkedlist *right,*left;
};

typedef struct linkedlist node;

void main()
{
node *start,*end;
int choice;
int menu(void);
node *create(node **lastnode);
void display(node *first,node *last);
void insert(node **first,node **last);
void del(node **first,node **last);
clrscr();
printf("\n DOUBLY LINKED LIST");
printf("\n ********************");
do
{
printf("\n\nMain menu");
printf("\n\n1.Create \n2.Insert \n3.Delete \n4.Display \n5.Exit");
choice =menu();

switch(choice)
{

case 1:
printf("\n Enter the data(-999 to stop):");
start=create(&end);
continue;

case 2:
insert(&start,&end);
```

```
printf("\n");
continue;

case 3:
del(&start,&end);
printf("\n");
continue;

case 4:
display(start,end);
printf("\n");
continue;

case 5:
   exit(0);
default:
  printf("\n\nINVALID CHOICE...");
}
}while(1);
}

int menu()
{
int choice;
do
{
printf("\n Enter your choice:");
scanf("%d",&choice);
if(choice<1||choice>5)
printf("\n Wrong choice");
}while(choice<1||choice>5);
printf("\n");
return(choice);
}

node *create(node **lastnode)
{
node *temp,*firstnode;
int info;
*lastnode=NULL;
firstnode=NULL;
scanf("%d",&info);
while(info!=-999)
```

```
{
temp=(node *)malloc(sizeof(node));
temp->item=info;
temp->right=NULL;
if(firstnode==NULL)
{
temp->left=NULL;
firstnode=temp;
}
else
{
temp->left=(*lastnode);
(*lastnode)->right=temp;
}
(*lastnode)=temp;
scanf("%d",&info);
}
if(firstnode!=NULL)
(*lastnode)=temp;
return(firstnode);
}

void display(node *first,node *last)
{
printf("\n Forward traversal\n");
while(first!=NULL)
{
printf("%d\t",first->item);
first=first->right;
}
printf("\n Backward traversal\n");
while(last!=NULL)
{
printf("%d\t",last->item);
last=last->left;
}
return;
}

void insert(node **first,node **last)
{
node *newnode;
int newitem;
```

```
int position;
node *temp;
int i;
printf("\n New data item:");
scanf("%d",&newitem);
do
{
printf("\n Position of insertion:");
scanf("%d",&position);
}while(position<=0);
if(((*first)==NULL)||(position==1))
{
newnode=(node *)malloc(sizeof(node));
newnode->item=newitem;
newnode->right=*first;
newnode->left=NULL;
if((*first)!=NULL)
(*first)->left=newnode;
else
(*last)=newnode;
*first=newnode;
}
else
{
i=1;
temp=*first;
while((i<position-1)&&(temp->right!=NULL))
{
i++;
temp=temp->right;
}
newnode=(node *)malloc(sizeof(node));
newnode->item=newitem;
newnode->right=temp->right;
if(temp->right!=NULL)
temp->right->left=newnode;
newnode->left=temp;
temp->right=newnode;
}
if(newnode->right==NULL)
*last=newnode;
}
```

```
void del(node **first,node **last)
{
node *temp,*prev;
int target;
printf("\n Enter the data to be deleted:");
scanf("%d",&target);
if(*first==NULL)
printf("\n List is empty");
else if((*first)->item==target)
{
if((*first)->right==NULL)
*first=*last=NULL;
else
{
*first=(*first)->right;
(*first)->left=NULL;
}
}
else
{
temp=*first;
prev=NULL;
while((temp->right!=NULL)&&(temp->item!=target))
{
prev=temp;
temp=temp->right;
}
if(temp->item!=target)
printf("\n Element not found");
else
{
if(temp==*last)
*last=prev;
else
temp->right->left=temp->left;
prev->right=temp->right;
}
}
}
```

-->>Sample Input Output:

```
Main menu

1.Create
2.Insert
3.Delete
4.Display
5.Exit
  Enter your choice:1

  Enter the data(-999 to stop):
5
10
15
-999

Main menu

1.Create
2.Insert
3.Delete
4.Display
5.Exit
  Enter your choice:2

New data item:20

Position of insertion:

Main menu

1.Create
2.Insert
3.Delete
4.Display
5.Exit
  Enter your choice:4

  Forward traversal
  5  20  10  15  20

  Backward traversal
 20  15  10  20  5
```

```
Main menu

1.Create
2.Insert
3.Delete
4.Display
5.Exit
   Enter your choice:3

  Enter the data to be deleted:5

Main menu

1.Create
2.Insert
3.Delete
4.Display
5.Exit
   Enter your choice:4

   Forward traversal
   20  10  15  20

   Backward traversal
  20  15  10  20
```