```
#include<stdio.h>
#include<stdlib.h>
/*Linked list declaration*/ struct node
{
int INFO;
struct node *link;
};
/*Declaring pointers to first and last node of the linked list*/ struct node *FIRST = NULL;
struct node *LAST = NULL;

/*Declaring function prototypes for linked list operations*/
void insert(int);
int delete(int);
void print(void);
struct node *search (int);

void main()
{
int num1, num2, choice;
struct node *location;

/*Displaying a menu of choices for performing linked list operations*/
while(1)
{

printf("\n\nSelect an option\n");
printf("\n1 - Insert");
printf("\n2 - Delete");
printf("\n3 - Search");
printf("\n4 - Print");
printf("\n5 - Exit");

printf("\n\nEnter your choice: ");
scanf("%d", &choice);

switch(choice)
{
case 1:
{
printf("\nEnter the element to be inserted into the linked list: ");
scanf("%d",&num1);
insert(num1); /*Calling the insert() function*/
printf("\n%d successfully inserted into the linked list!",num1);
break;
}

case 2:
{
printf("\nEnter the element to be deleted from the linked list: ");
scanf("%d",&num1);
num2=delete(num1); /*Calling the delete() function */
if(num2==-9999)
printf("\n\t%d is not present in the linked list\n\t",num1);
else
printf("\n\tElement %d successfuly deleted from the linked list\n\t",num2);
break;
}

case 3:
{
printf("\nEnter the element to be searched: ");
scanf("%d",&num1);
location=search(num1); /*Calling the search() function*/
if(location==NULL)
printf("\n\t%d is not present in the linked list\n\t",num1);
else
{
if(location==LAST)
printf("\n\tElement %d is the last element in the list",num1);
else
printf("\n\tElement %d is present before element %d in the linked list",num1,(location->LINK)-
>INFO);
}

break;
}

case 4:
{
print(); /*Printing the linked list elements*/

break;
}

case 5:
{
exit(1);
break;
}

default:
{
printf("\nIncorrect choice. Please try again.");
break;
}
}
}
}

/*Insert function*/

void insert(int value)
{
/*Creating a new node*/
struct node *PTR = (struct node*)malloc(sizeof(struct node));

/*Storing the element to be inserted in the new node*/
PTR->INFO = value;
```

```
/*Linking the new node to the linked list*/ if(FIRST==NULL)
{
FIRST = LAST = PTR;
PTR->LINK=NULL;
}
else
{
LAST->LINK = PTR;
PTR->LINK = NULL;
LAST = PTR;
}
}

/*Delete function*/
int delete(int value)
{
struct node *LOC,*TEMP; int i;
i=value;
LOC=search(i); /*Calling the search() function*/
if(LOC==NULL) /*Element not found*/
return(-9999);

if(LOC==FIRST)
{ if(FIRST==LAST) FIRST=LAST=NULL;
else
FIRST=FIRST->LINK;
return(value);
}

for(TEMP=FIRST;TEMP->LINK!=LOC;TEMP=TEMP->LINK);
TEMP->LINK=LOC->LINK; if(LOC==LAST)



LAST=TEMP;
return(LOC->INFO);
}

/*Search function*/
struct node *search (int value)
{
struct node *PTR;

if(FIRST==NULL) /*Checking for empty list*/ return(NULL);

/*Searching the linked list*/
for(PTR=FIRST;PTR!=LAST;PTR=PTR->LINK)
if(PTR->INFO==value)
return(PTR); /*Returning the location of the searched element*/

if(LAST->INFO==value) return(LAST);
else
return(NULL); /*Returning NULL value indicating unsuccessful search*/
}
```

```
/*print function*/ void print()
{
struct node *PTR;

if(FIRST==NULL) /*Checking whether the list is empty*/
{
printf("\n\tEmpty List!!"); return;
}

printf("\nLinked list elements:\n");
if(FIRST==LAST) /*Checking if there is only one element in the list*/
{
printf("\t%d",FIRST->INFO); return;
}

/*Printing the list elements*/
for(PTR=FIRST;PTR!=LAST;PTR=PTR->LINK)
printf("\t%d",PTR->INFO);
printf("\t%d",LAST->INFO);
}
```

Output:

```
Select an option

1 - Insert
2 - Delete
3 - Search
4 - Print
5 - Exit

Enter your choice: 111

Incorrect choice. Please try again.

Select an option

1 - Insert
2 - Delete
3 - Search
4 - Print
5 - Exit

Enter your choice: 1

Enter the element to be inserted into the linked list: 10

10 successfully inserted into the linked list!

Select an option

1 - Insert
2 - Delete
3 - Search
```

```
4 - Print
5 - Exit

Enter your choice: 1

Enter the element to be inserted into the linked list: 20

20 successfully inserted into the linked list!

Select an option

1 - Insert
2 - Delete
3 - Search
4 - Print
5 - Exit

Enter your choice:
1

Enter the element to be inserted into the linked list: 30

30 successfully inserted into the linked list!

Select an option

1 - Insert
2 - Delete
3 - Search
4 - Print
5 - Exit

Enter your choice: 4

Linked list elements:
        10      20      30

Select an option

1 - Insert
2 - Delete
3 - Search
4 - Print
5 - Exit

Enter your choice: 3

Enter the element to be searched: 20

        Element 20 is present before element 30 in the linked list

Select an option

1 - Insert
2 - Delete
```

```
3 - Search
4 - Print
5 - Exit

Enter your choice: 3

Enter the element to be searched: 50

        50 is not present in the linked list


Select an option

1 - Insert
2 - Delete
3 - Search
4 - Print
5 - Exit

Enter your choice: 2

Enter the element to be deleted from the linked list: 20

        Element 20 successfuly deleted from the linked list


Select an option

1 - Insert
2 - Delete
3 - Search
4 - Print
5 - Exit

Enter your choice: 4

Linked list elements:
        10      30

Select an option

1 - Insert
2 - Delete
3 - Search
4 - Print
5 - Exit

Enter your choice: 5
```