Code:

```
#include <stdio.h>
#include<stdlib.h>

struct list{
      int info;
      struct list *pre, *next;
};

typedef struct list node;

node *start=NULL, *last=NULL;

void f_insert();
void e_insert();
void m_insert();

void f_delete();
void e_delete();
void m_delete();

void traverse();
void back_traverse();
void search();
void destroy();

void main() {
    int ch;
    printf("\n\n\n\t 1. Insert at first ");
    printf("\n\t 2. Insert at last");
    printf("\n\t 3. Insert at specified position ");
    printf("\n\t 4. Delete from first ");
    printf("\n\t 5. Delete from last ");
    printf("\n\t 6. Delete from specific position");
    printf("\n\t 7. Display all values from beginning");
    printf("\n\t 8. Display all values from last");
    printf("\n\t 9. Search an item");
    printf("\n\t 10. Delete list");
    printf("\n\t 11. Exit");
    while(1){
        printf("\n Enter your choice: ");
        scanf("%d", &ch);
        switch(ch){
        case 1:
            f_insert();
            break;
        case 2:
            e_insert();
            break;
        case 3:
            m_insert();
            break;
        case 4:
            f_delete();
            break;
```

```
        case 5:
            e_delete();
            break;
        case 6:
            m_delete();
            break;
        case 7:
            traverse();
            break;
        case 8:
            back_traverse();
            break;
        case 9:
            search();
            break;
        case 10:
            destroy();
            break;
        case 11:
            printf("\n\n\t Program terminated successfully !!!");
            break;
        default:
            printf("\t\t Please enter the correct choice!!! \n");
        }
        if(ch==11)
            break;
    }
}
void f_insert(){
      node *item = (node*)malloc(sizeof(node));
    printf("Enter data to input: ");
    scanf("%d", &item->info);
    item->pre=NULL;
    if(start==NULL){
        item->next=NULL;
        start=item;
        last=item;
    }
    else{
        item->next=start;
        start->pre=item;
        start=item;
    }
    printf("Node inserted");
}
void e_insert(){
      node *item = (node*)malloc(sizeof(node));
    printf("Enter data to input: ");
    scanf("%d", &item->info);
    item->next=NULL;
    if(start==NULL){
      item->pre=NULL;
        start=item;
        last=start;
    }
    else{
```

```
      item->pre=last;
      last->next=item;
      last=item;
    }
    printf("Node inserted");
}
void m_insert(){
    int n, count=1;
    node *item, *temp;
    item=(node*)malloc(sizeof(node));
    printf("Enter the position : ");
    scanf("%d", &n);
    printf("Enter data to input : ");
    scanf("%d", &item->info);
    temp=start;
      if(n==1){
      item->next=start;
      start->pre=item;
      start=item;
      start->pre=NULL;
      printf("Node inserted");
    }
    else{
      while (temp->next!=NULL){
            if(count==(n-1))
                  break;
             temp=temp->next;
             count++;
        }
            if(count==(n-1)){
                  item->pre=temp;
                  item->next=temp->next;
                  if(temp->next!=NULL)
                        temp->next->pre=item;
                  else
                        last=item;
                  temp->next=item;
                  printf("Node inserted");
            }
            else
                  printf("Position undefined");
      }
}

void f_delete(){
      node *ptr;
      if(start==NULL)
            printf("Empty list");
      else if(start->next==NULL){
            printf("The deleted data is: %d", start->info);
            free(start);
            start=NULL;
      }
    else{
        ptr=start;
        start=start->next;
```

```
        start->pre=NULL;
           printf("The deleted data is : %d", ptr->info);
        free(ptr);
    }
}
void e_delete(){
      node *ptr, *loc;
    if(start==NULL)
        printf("Empty list");
    else if(start->next==NULL){
        printf("Deleted data is : %d", start->info);
        free(start);
        start=NULL;
    }
    else{
      node *ptr=last;
      last=last->pre;
        last->next = NULL;
        printf("The deleted data is: %d", ptr->info);
        free(ptr);
    }
}
void m_delete(){
    node *ptr, *temp;
      int n, count=0;
    if(start==NULL)
        printf("Empty list");
    else{
        printf("Enter the position : ");
        scanf("%d", &n);
        ptr=start;
        if((start->next==NULL) && (n==1)){
           printf("The deleted data is: %d", start->info);
           free(start);
           start=NULL;
        }
           else if(n==1){
                 ptr=start;
                 start=start->next;
                 start->pre=NULL;
                 printf("The deleted data is : %d", ptr->info);
                 free(ptr);
           }
        else{
           while(ptr != NULL){
                       count++;
                       if(count==n)
                             break;
                       temp=ptr;
                 ptr=ptr->next;
            }
            if(count==n){
                 temp->next=ptr->next;
                 if(ptr->next!=NULL)
                       ptr->next->pre=temp;
                 else
```

```
                    last=ptr->pre;
                printf("Deleted data is: %d", ptr->info);
                free(ptr);
            }
            else
                printf("Invalid position");
        }
    }
}

void traverse(){
      node *temp;
    temp=start;
    if(temp==NULL)
        printf("Empty list");
    else{
      printf("Elements of list in usual order are :- ");
      while (temp != NULL){
            printf("\n\t %d", temp->info);
            temp=temp->next;
        }
    }
}
void back_traverse(){
      node *temp;
    temp=last;
    if(temp==NULL)
        printf("Empty list");
    else{
      printf("Elements of list in reverse order are :- ");
      while (temp!= NULL){
            printf("\n\t %d", temp->info);
            temp=temp->pre;
        }
    }
}
void search(){
      node *temp;
    int key, count=1;
    printf("Enter element to search : ");
    scanf("%d", &key);
    temp=start;
    while(temp != NULL){
        if(temp->info==key)
            break;
        else{
            count++;
            temp=temp->next;
        }
    }
    if(start==NULL)
        printf("List is empty");
    else if(temp==NULL)
        printf("Element not found");
    else
        printf("Element found and the position is %d", count);
```

```
}
void destroy(){
      node *temp;
    if(start==NULL)
        printf("Empty list");
    else{
        while (start != NULL){
            temp=start;
            start=start->next;
            free(temp);
        }
        printf("List destroyed");
    }
}
```

Output:

1. Insert at first
2. Insert at last
3. Insert at specified position
4. Delete from first
5. Delete from last
6. Delete from specific position
7. Display all values from beginning
8. Display all values from last
9. Search an item
10. Delete list
11. Exit

Enter your choice: 1
Enter data to input: 2
Node inserted
Enter your choice: 2
Enter data to input: 5
Node inserted
Enter your choice: 2
Enter data to input: 9
Node inserted
Enter your choice: 7
Elements of list in usual order are :-

2
5

```
        9
 Enter your choice: 8
Elements of list in reverse order are :-
        9
        5
        2
 Enter your choice: 3
Enter the position : 5
Enter data to input : 88
Position undefined
 Enter your choice: 3
Enter the position : 4
Enter data to input : 88
Node inserted
 Enter your choice: 7
Elements of list in usual order are :-
        2
        5
        9
        88
 Enter your choice: 8
Elements of list in reverse order are :-
        88
        9
        5
        2
 Enter your choice: 6
Enter the position : 0
Invalid position
 Enter your choice: 6
Enter the position : 2
Deleted data is: 5
 Enter your choice: 7
Elements of list in usual order are :-
        2
        9
        88
 Enter your choice: 8
Elements of list in reverse order are :-
        88
        9
        2
 Enter your choice: 9
Enter element to search : 88
Element found and the position is 3
 Enter your choice: 9
Enter element to search : 5
Element not found
```

```
 Enter your choice: 4
The deleted data is : 2
 Enter your choice: 7
Elements of list in usual order are :-
        9
        88
 Enter your choice: 8
Elements of list in reverse order are :-
        88
        9
 Enter your choice: 3
Enter the positon : 1
Enter data to input : 55
Node inserted
 Enter your choice: 7
Elements of list in usual order are :-
        55
        9
        88
 Enter your choice: 8
Elements of list in reverse order are :-
        88
        9
        55
 Enter your choice: 5
The deleted data is: 88
 Enter your choice: 7
Elements of list in usual order are :-
        55
        9
 Enter your choice: 8
Elements of list in reverse order are :-
        9
        55
 Enter your choice: 100
                Please enter the correct choice!!!

 Enter your choice: 10
List destroyed
 Enter your choice: 4
Empty list
 Enter your choice: 11


        Program terminated successfully !!!
--------------------------------
Process exited with return value 11
Press any key to continue . . .
```