```
#include<stdio.h>
#include<stdlib.h>
struct node
{
   int data;
   struct node *next;
};
struct node *start;
            /*fuction declaration of all the operations*/
void insert_begin();
void insert_last();
void insert_locc();
void delete_begin();
void delete_last();
void delete_locc();
void print();
void main ()
{
   int ch=0;
   while(ch!=8)
   {
      printf("\nEnter the operation to be performed\n");
      printf("\n1.Insert in the begining\n2.Insert at last\n3.Insert at any specified position\n4.Delete
from Beginning\n5.Delete from last\n6.Delete node after specified location\n7.Show\n8.Exit\n");
      scanf("\n%d",&ch);
      switch(ch)
      {      /*function calls of all the operations */
         case 1:
         insert_begin();
         break;
         case 2:
```

```
            insert_last();
            break;
            case 3:
            insert_locc();
            break;
            case 4:
            delete_begin();
            break;
            case 5:
            delete_last();
            break;
            case 6:
            delete_locc();
            break;
            case 7:
            print();
            break;
            case 8:
            exit(0);
            break;
            default:
            printf("Enter valid option");
        }
    }
}       /*function definition*/
void insert_begin()             //to insert the node at the beginnning of linked list
{
    struct node *p;
    int value;
    p=(struct node *) malloc(sizeof(struct node *));
    if(p==NULL)
```

```
    {
        printf("\nOVERFLOW");
    }
    else
    {
        printf("\nEnter value\n");
        scanf("%d",&value);
        p->data=value;
        p->next=start;
        start=p;
    }
}
void insert_last()          //to insert the node at the last of linked list
{
    struct node *p,*temp;
    int value;
    p=(struct node*)malloc(sizeof(struct node));
    if(p==NULL)
    {
        printf("\nOVERFLOW");
    }
    else
    {
        printf("\nEnter value\n");
        scanf("%d",&value);
        p->data=value;
        if(start==NULL)
        {
            p->next=NULL;
            start=p;
        }
```

```
        else
        {
            temp=start;
            while(temp->next!=NULL)
            {
                temp=temp->next;
            }
            temp->next=p;
            p->next=NULL;
        }
    }
}
void insert_locc()              //to insert the node at the specified location of linked list
{
    int i,loc,value;
    struct node *p, *temp;
    p=(struct node *)malloc(sizeof(struct node));
    if(p==NULL)
    {
        printf("\nOVERFLOW");
    }
    else
    {
        printf("\nEnter element value");
        scanf("%d",&value);
        p->data=value;
        printf("\nEnter the location after which you want to insert ");
        scanf("\n%d",&loc);
        temp=start;
        for(i=0;i<loc;i++)
        {
```

```
            temp=temp->next;
            if(temp==NULL)
            {
               printf("\ncan't insert\n");
               return;
            }
         }
         p->next=temp->next;
         temp->next=p;
      }
}
void delete_begin()        //to delete the node present in the beginning of the linked list
{
   struct node *p;
   if(start==NULL)
   {
      printf("\nList is empty\n");
   }
   else
   {
      p=start;
      start=p->next;
      free(p);
   }
}
void delete_last()        //to delete the node present in the last of the linked list
{
   struct node *p,*p1;
   if(start==NULL)
   {
      printf("\nlist is empty");
```

```
    }
    else if(start->next==NULL)
    {
        start=NULL;
        free(start);
        printf("\nOnly node of the list deleted ...\n");
    }
    else
    {
        p=start;
        while(p->next!=NULL)
        {
            p1=p;
            p=p->next;
        }
        p1->next=NULL;
        free(p);
    }
}
void delete_locc()    //to delete the node present at the specified of the linked list
{
    struct node *p,*p1;
    int loc,i;
    printf("\n Enter the location of the node after which you want to perform deletion \n");
    scanf("%d",&loc);
    p=start;
    for(i=0;i<loc;i++)
    {
        p1=p;
        p=p->next;
```

```
        if(p==NULL)
        {
            printf("\nCan't delete");
            return;
        }
    }
    p1->next=p->next;
    free(p);
    printf("\nDeleted node %d ",loc+1);
}
void print()    //to print the values in the linked list
{
    struct node *p;
    p=start;
    if(p==NULL)
    {
        printf("Nothing to print");
    }
    else
    {
        printf("\nprinting values\n");
        while (p!=NULL)
        {
            printf("\n%d",p->data);
            p=p->next;
        }
    }
}
```

```
Enter the operation to be performed

1.Insert in the begining
2.Insert at last
3.Insert at any specified position
4.Delete from Beginning
5.Delete from last
6.Delete node after specified location
7.Show
8.Exit
1

Enter value
89

Enter the operation to be performed

1.Insert in the begining
2.Insert at last
3.Insert at any specified position
4.Delete from Beginning
5.Delete from last
6.Delete node after specified location
7.Show
8.Exit
8
```