```
#include<stdio.h>
#include<stdlib.h>
struct node
{
    int data;
    struct node *next;
};
struct node *head;
                /*function declaration of all the operations*/
void create();
void insert_begin();
void insert_end();
void insert_middle();
void delete_begin();
void delete_end();
void delete_middle();
void print();
void main ()
{
    int ch=0;
    while(ch!=9)
    {
        printf("\nEnter the operation to be performed\n");
        printf("\n1.Create the list\n2.Insert in the begining\n3.Insert at last\n4.Insert at
any specified position\n5.Delete from Beginning\n6.Delete from last\n7.Delete
node after specified location\n8.Show\n9.Exit\n");
}

   scanf("\n%d",&ch);
   switch(ch)
   {       /*function calls of all the operations */
      case 1:  create();
      break;
      case 2:  insert_begin();
      break;
      case 3:  insert_end();
      break;
      case 4:  insert_middle();
      break;
      case 5:  delete_begin();
      break;
      case 6:  delete_end();
      break;
      case 7:  delete_middle();
      break;
      case 8:  print();
      break;
      case 9:  exit(0);
      break;
      default:
      printf("Enter valid option");
   }
}
```

```
}
/*function definition*/
void create(){
    struct node *temp,*new_node;
        new_node=(struct node *)malloc(sizeof(struct node));
        if(new_node==NULL)
        {
                printf("nOut of Memory Space:n");
                exit(0);
        }
        printf("Enter the data value for the node:");
        scanf("%d",&new_node->data);
        new_node->next=NULL;
        if(head==NULL)
        {
                head=new_node;
        }
        else
        {
                temp=head;
                while(temp->next!=NULL)
                {
                        temp=temp->next;
                }
                temp->next=new_node;
        }
}

void insert_begin()                  //to insert the node at the beginning 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=head;
        head=p;
    }
}
```

```
void insert_end()              //to insert the node at the end of the 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(head==NULL)
      {
         p->next=NULL;
         head=p;
      }
      else
      {
         temp=head;
         while(temp->next!=NULL)
         {
            temp=temp->next;
            }
            temp->next=p;
            p->next=NULL;
        }
    }
}
void insert_middle()                //to insert the node at the specified location of linked
list
{
    int i,loc,value;
    struct node *new_node, *temp;
    new_node=(struct node *)malloc(sizeof(struct node));
    if(new_node==NULL)
    {
        printf("\nOVERFLOW");
    }
    else
    {
        printf("\nEnter element value");
        scanf("%d",&value);
        new_node->data=value;
        printf("\nEnter the location after which you want to insert ");
        scanf("\n%d",&loc);
        temp=head;
        for(i=0;i<loc;i++)
        {
            temp=temp->next;
            if(temp==NULL)
            {
                printf("\ncan't insert\n");
                return;
            }
        }
        new_node->next=temp->next;
        temp->next=new_node;
    }
}
void delete_begin()    //to delete the node present in the beginning of the linked list
{
    struct node *temp;
    if(head==NULL)
    {
        printf("\nList is empty\n");
    }
    else
    {
        temp=head;
        head=temp->next;
        free(temp);
    }
}
```

```
void delete_end()          //to delete the node present in the last of the linked list
{
   struct node *temp,*prev_node;
   if(head==NULL)
   {
      printf("\nlist is empty");
   }
   else if(head->next==NULL)
   {
      head=NULL;
      free(head);
      printf("\nOnly node of the list deleted ...\n");
   }
   else
   {
      temp=head;
      while(temp->next!=NULL)
      {
         prev_node=temp;
         temp=temp->next;
      }
      prev_node->next=NULL;
      free(temp);
   }
}
```

```
void delete_middle()    //to delete the node present at the specified of the linked list
{
   struct node *temp,*prev_node;
   int loc,i;
   printf("\n Enter the location of the node after which you want to perform
deletion \n");
   scanf("%d",&loc);
   temp=head;
   for(i=0;i<loc;i++)
   {
      prev_node=temp;
      temp=temp->next;

      if(temp==NULL)
      {
         printf("\nCan't delete");
         return;
      }
   }
   prev_node->next=temp->next;
   free(temp);
   printf("\nDeleted node %d ",loc+1);
}
void print()    //to print the values in the linked list
{
   struct node *p;
```

```
    p=head;
    if(p==NULL)
    {
        printf("Nothing to print");
    }
    else
    {
        printf("\nprinting values\n");
        while (p!=NULL)
        {
            printf("\n%d",p->data);
            p=p->next;
        }
    }
}
```