```
/*INSERTION AND DELETION IN SINGLY LINKED LIST*/
#include<stdio.h>
#include<conio.h>
void Insertion_at_beg();
void Insertion_at_end();
void Insertion_at_specific();
void Deletion_from_beg();
void Deletion_from_end();
void Deletion_from_specific();
void display();

struct node
{
int data;
struct node *next;
}

void main()
{
int ch;
clrscr();
printf("1.Insertion at beg \n
        2.Insertion at end \n
        3.Insertion at specific position
        4.Deletion from beg
```

```
        5.Deletion from end
        6.Deletion from specific
        7.Display\n
        8.Exit");
do
{
printf("\nEnter ur choice: ");
scanf("%d",&ch);
switch(ch)
{
case 1:Insertion_at_beg();break;
case 2:Insertion_at_end();break;
case 3:Insertion_at_specific();break;
case 4:Deletion_from_beg();break;
case 5:Deletion_from_end();break;
case 6:Deletion_from_specific;break;
case 7:Display();break;
case 8:printf("Program Exited");break;
default:printf("Invalid choice");
}
}while(ch!=8);
getch();
}
```

```
void Insertion_at_beg()
{
struct node *ptr;
printf("Enter the item");
scanf("%d",&item);
if(start==NULL)
{
start=(struct node *)malloc(sizeof(struct node));
start->data=item;
start->next=NULL;
}
{
ptr=(struct node *)malloc(sizeof(struct node));
ptr->data=item;
ptr->next=start;
start=ptr;
}
printf("Inserted Element is %d",item);
}
void Insertion_at_end()
{
struct node *ptr,*temp;
printf("Enter the item");
scanf("%d",&item);
if(start==NULL)
{
```

```
start=(struct node *)malloc(sizeof(struct node));
start->data=item;
start->next=NULL;
}
else
{
ptr=(struct node *)malloc(sizeof(struct node));
Ptr->data=item;
Ptr->next=NULL;
Temp=start;
while(temp->next!=NULL)
{
temp=temp->next;
}
temp->next=ptr;
}
void insert_specific()
 {
        int n;
        struct node *nw, *ptr;
if (start == NULL)
 printf("\n\nLinked list is empty. It must have at least one node.\n");
 else
 {
        printf("\n\nEnter INFO after which new node is to be inserted: ");
        scanf("%d", &n);
```

```
        printf("\n\nEnter ITEM: ");
        scanf("%d", &item);
        ptr = start;
        nw = start;
 while (ptr-> != NULL)
        {
        if (ptr->data == n)
                {
                nw = (struct node *)malloc(sizeof(struct node));
                nw->data = item;
                nw->next = ptr->next;
                ptr->next = nw;
                printf("\n\nItem inserted: %d", item); return;

        }
 else
 ptr = ptr->next;
}
}
}
void delete_beg()
{
 int x;
struct node*ptr;
if(start==NULL)
printf("Linked is empty");
```

```
else
{
 ptr=start;
x=ptr->data;
start=p->next;
printf("The deleted element is  %d",x);
free(ptr);
}
}
void delete_end()
{
struct node *ptr,*temp;
int x;
ptr=start;
if(ptr==NULL)
printf("Linked list is empty");
else
{
if(ptr->next==NULL)
{
 x=ptr->data;
start=NULL;
}
else
{
while(ptr->next!=NULL)
```

```
{
 temp=ptr;
ptr=ptr->next;
}
x=ptr->data;
temp->next=NULL;
}
printf("Deleted element %d",x);
free(ptr);
}

Void delete_any()
{
int key;
if(start->next==NULL)
      printf("\n Empty Linked list. Deletion not possible.");
else
{
printf("Enter data of node to be deleted.");
scanf("%d",&key);
ptr=start;
while((ptr->next !=NULL)||(prt->data !=key))
{
ptr1=ptr;
ptr=ptr->next;
}
```

```
if(ptr->data = = key)
{
ptr1->next=ptr->next;
free(ptr);
printf("\n node with data %d deleted.",key);
}
else
printf("no such data found");
}
}
Void display()
{
Struct node *ptr;
Ptr=start;
If (ptr==NULL)
printf("Linked list is empty");
else
{
While (ptr!=NULL)
{
Printf("Items in Linked list are%d",ptr->data);
Ptr=ptr->next;
}
}
}
```