```
#include <stdio.h>
#include <stdlib.h>
typedef struct node{
    int data.
    stud node *link;
}node;
node *root = NULL;
void add_at_end()
{ node *temp;
  temp = (node *) malloc (sizeof (node));
  printf ("Enter the node element \n");
  scanf ("%d", &temp->data);
  temp -> link = NULL.
if (root == NULL)
  { root = temp;
}
else
  { node *p = root;
  while (p->link != NULL)
    { p = p -> link;
    }
    p -> link = temp;
  }
}
```

```
void add_at_begin ()
{
  node *temp;
temp = (node*) malloc(sizeof(node));
printf("Enter node element \n");
scanf("%d", &temp->data);
temp->link = null;
if (root == NULL)
  {
  root = temp;
}
else
  {
  temp->link = root;
  root = temp;
}
}

int length()
{
node *p;
  p = root;
  int i = 0;
  while (p != NULL)
  {
  i++;
   p = p->link;
}
```

```
return L;
}
void add_after ()
{
 node *p, *temp;
 int loc, i=1;
 printf ("Enter the location ");
 scanf ("%d", &loc);
 if (loc > length())
 {
 print ("Invalid location, the list has %d nodes", length());
}
else
 {
 p=root;
 while (i<loc)
 {
  p=p->link;
  i++;
 }
temp = (node *)malloc (sizeof (node));
printf ("Enter the node element \n");
scan f ("%d", &temp->data);
 temp->link=NULL;
 temp->link = p->link;
 p->link = temp;
 }
 }
```

```
void delete()
{
 int loc;
 node *temp;
 printf("Enter the location of node to be deleted\n");
 scanf("%d",&loc);
  if (loc > length())
 {
 printf("There is no such node\n");
 }
 else if (loc == 1)
 {
   temp = root;
   root = temp->link;
   temp->link = null;
   free(temp);
   }
   else
   {
   node *p=root, *q;
   int i=1;
   while (i<loc-1)
    {
      p = p->link;
     i++;
 }
   q = p->link;
   p->link = q->link;
  q->Link = NULL;
  free(q);
  }
  }
```

```
void display()()
{
node *temp=root;
if (temp==NULL)
{printf("NO nodes in the list \n");
}
else
{while(temp!=NULL)
  {
  printf("%d \n", temp -> data );
  temp=temp ->link;
 }
}
}
int main()
{
int op, len;
while(1)
{printf("Enter the Operation \n1.Add in begin \n2 add
        at end\n");
printf("3.Add after a node\n 4. Delete node\n 5. Display
 \n 6. length of list \n7. Exit \n");
 scanf("%d",&op);
 swith(op)
{
case 1: add_at_begin();
        break;
case 2: add_at_end();
        break;
```

```
case 3 : add-after();
        break.
case 4 : delete();
        break;
case 5 : display();
    break;
case 6 : len = length().
            printf("the length is %d \n", len);
        break;
case 7 : exit(0)
        break;
default : printf("No such Operation\n");
 }
 }
return 0;
}.
```