LINEAR LINKED LIST

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

typedef struct node{
 int data;
 struct 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++;
```

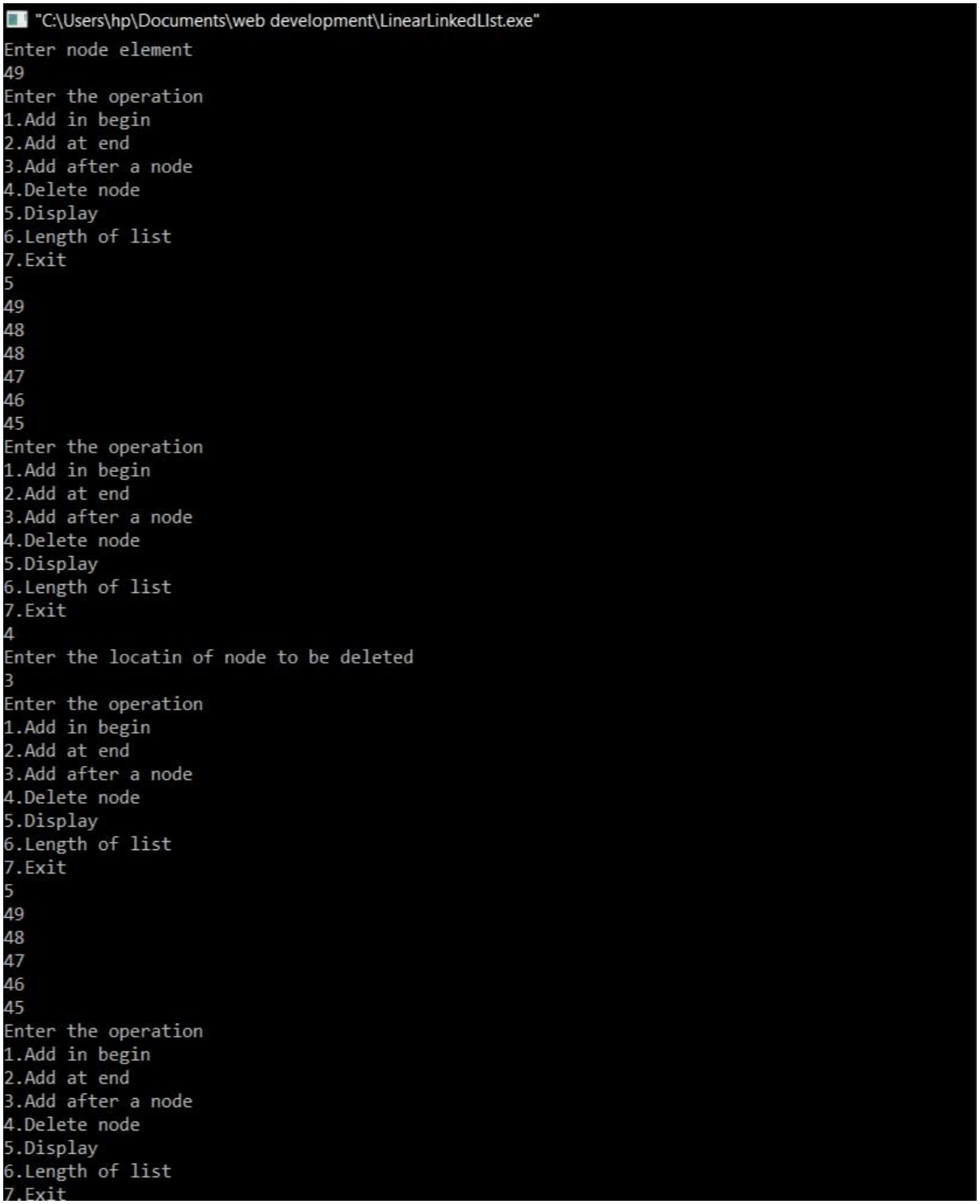
```
   p=p->link;
  }
return i;
}

void add_after(){

node *p,*temp;
int loc,i=1;
printf("Enter the location");
scanf("%d",&loc);

if(loc>length())
{
  printf("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");
scanf("%d",&temp->data);
temp->link=NULL;
```

```
temp->link=p->link;
p->link=temp;
}
}

void delete()
{
int loc;
node *temp;
printf("Enter the locatin 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\n4.Delete node\n5.Display\n6.Length of list\n7.Exit\n");
  scanf("%d",&op);
  switch (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;
}
```

OUTPUT


"C:\Users\hp\Documents\web development\LinearLinkedLIst.exe"
Enter node element
49
Enter the operation
1.Add in begin
2.Add at end
3.Add after a node
4.Delete node
5.Display
6.Length of list
7.Exit
5
49
48
48
47
46
45
Enter the operation
1.Add in begin
2.Add at end
3.Add after a node
4.Delete node
5.Display
6.Length of list
7.Exit
4
Enter the locatin of node to be deleted
3
Enter the operation
1.Add in begin
2.Add at end
3.Add after a node
4.Delete node
5.Display
6.Length of list
7.Exit
5
49
48
47
46
45
Enter the operation
1.Add in begin
2.Add at end
3.Add after a node
4.Delete node
5.Display
6.Length of list
7.Exit