```
/* Program to implement LINKED LIST:
insert-front, delete rear, display, count the items , Search the items, order the list */

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

typedef struct node{
 int data;
 struct node *link;
}node;

node *root=NULL;
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;
 }
}
void delete_rear()
```

```
{
  // struct node* temp=root;
   struct node *ptr1,*ptr2;
   ptr1=root;
   while(ptr1->link != NULL)
   {
      ptr2=ptr1;
      ptr1=ptr1->link;
   }
   ptr2->link=NULL;
   printf("item delted is %d ",ptr1->data);
   free(ptr1);
return;
}
int length()
{
 node *p;
 p=root;
 int i=0;

 while(p!=NULL)
 {
  i++;
  p=p->link;
 }
return i;
}
void sortList(){
      //Node current will point to head
      struct node *current = root, *index = NULL;
      int temp;
```

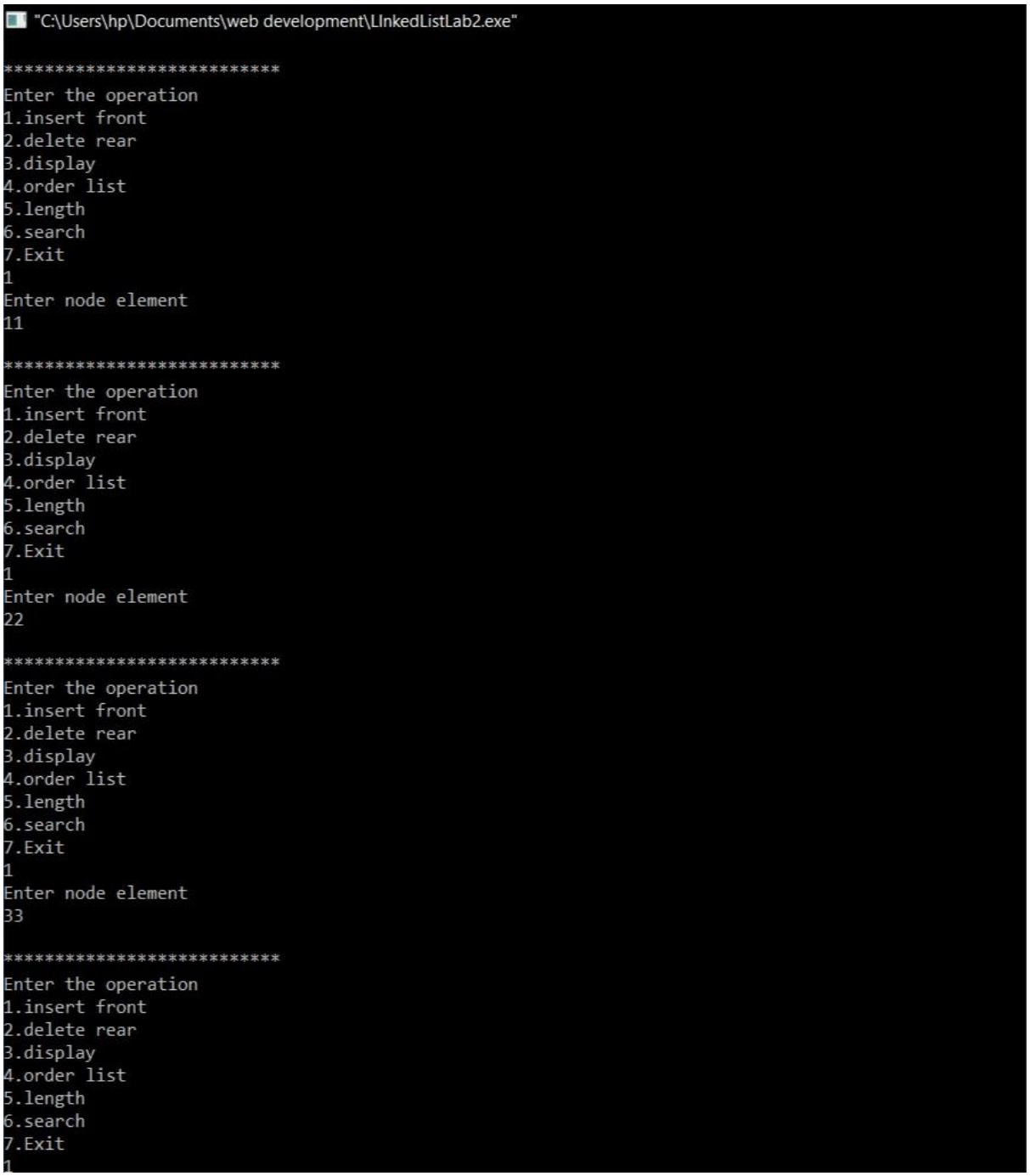
```
    if(root == NULL) {
       return;
    }
    else {
       while(current != NULL) {
          //Node index will point to node next to current
          index = current->link;

          while(index != NULL) {
             //If current node's data is greater than index's node data, swap the data between them
             if(current->data > index->data) {
                temp = current->data;
                current->data = index->data;
                index->data = temp;
             }
             index = index->link;
          }
          current = current->link;
       }
    }
}
void search()
{
   struct node *temp=root;
   int key;
   int ct=0;
   int flag=0;
   printf("Enter the element need to be searched:");
   scanf("%d",&key);
   while(temp != NULL)
```

```
    {

        ct++;
        if(temp->data==key)
        {
           flag=1;
           printf("element %d is there in position at %d ",key,ct);
        }
       temp=temp->link;
    }
    if(flag==0)
    printf("element is not there in list\n");
    return;
}
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("\n**************************\nEnter the operation\n1.insert front\n2.delete rear\n");
  printf("3.display\n4.order list\n5.length\n6.search\n7.Exit\n");
  scanf("%d",&op);
  switch (op)
  {
  case 1:add_at_begin();
        break;
  case 2: delete_rear();
        break;
  case 3: display();
        break;
  case 4: sortList();
        break;

  case 5: len=length();
        printf("The length is %d\n",len);
        break;
  case 6: search();
        break;
  case 7:exit(0);
        break;

  default: printf("No such operation\n");
  }
 }
```

return 0;

}


"C:\Users\hp\Documents\web development\LInkedListLab2.exe"
****
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
1
Enter node element
11
****
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
1
Enter node element
22
****
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
1
Enter node element
33
****
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
1

```
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
1
Enter node element
55

**************************
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
3
55
44
33
22
11

**************************
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
6
Enter the element need to be searched:1000
element is not there in list

**************************
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
```

```
element is not there in list

*************************
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
6
Enter the element need to be searched:55
element 55 is there in position at 1
*************************
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
5
The length is 5

*************************
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
2
item delted is 11
*************************
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
3
55
44
33
```

```
**************************
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
4

**************************
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
3
22
33
44
55

**************************
Enter the operation
1.insert front
2.delete rear
3.display
4.order list
5.length
6.search
7.Exit
```