```
#include<stdio.h>
#include <stdlib.h>
struct node
{
  int info;
  struct node *link;
};
typedef struct node *NODE;

NODE getnode()
{
NODE x;
x=(NODE)malloc(sizeof(struct node));
if(x==NULL)
 {
  printf("mem full\n");
  exit(0);
 }
 return x;
}

void freenode(NODE x)
{
free(x);
}

NODE insert_front(NODE first,int item)
{
NODE temp;
```

```
temp=getnode();
temp->info=item;
temp->link=NULL;
if(first==NULL)
return temp;
temp->link=first;
first=temp;
return first;
}

NODE delete_rear(NODE first)
{
NODE cur,prev;
if(first==NULL)
{
printf("list is empty cannot delete\n");
return first;
}
if(first->link==NULL)
{
printf("item deleted is %d\n",first->info);
free(first);
return NULL;
}
prev=NULL;
cur=first;
while(cur->link!=NULL)
{
prev=cur;
```

```
cur=cur->link;
}
printf("iten deleted at rear-end is %d",cur->info);
free(cur);
prev->link=NULL;
return first;
}

void display(NODE first)
{
 NODE temp;
 if(first==NULL)
 printf("list empty cannot display items\n");
 for(temp=first;temp!=NULL;temp=temp->link)
  {
  printf("%d\n",temp->info);
  }
}

void count(NODE first)
{
        NODE temp;
        int count=1;
        while(temp->link!=NULL)
        {
                temp=temp->link;
                count=count+1;
        }
        printf("The length of linked list is: %d",count);
```

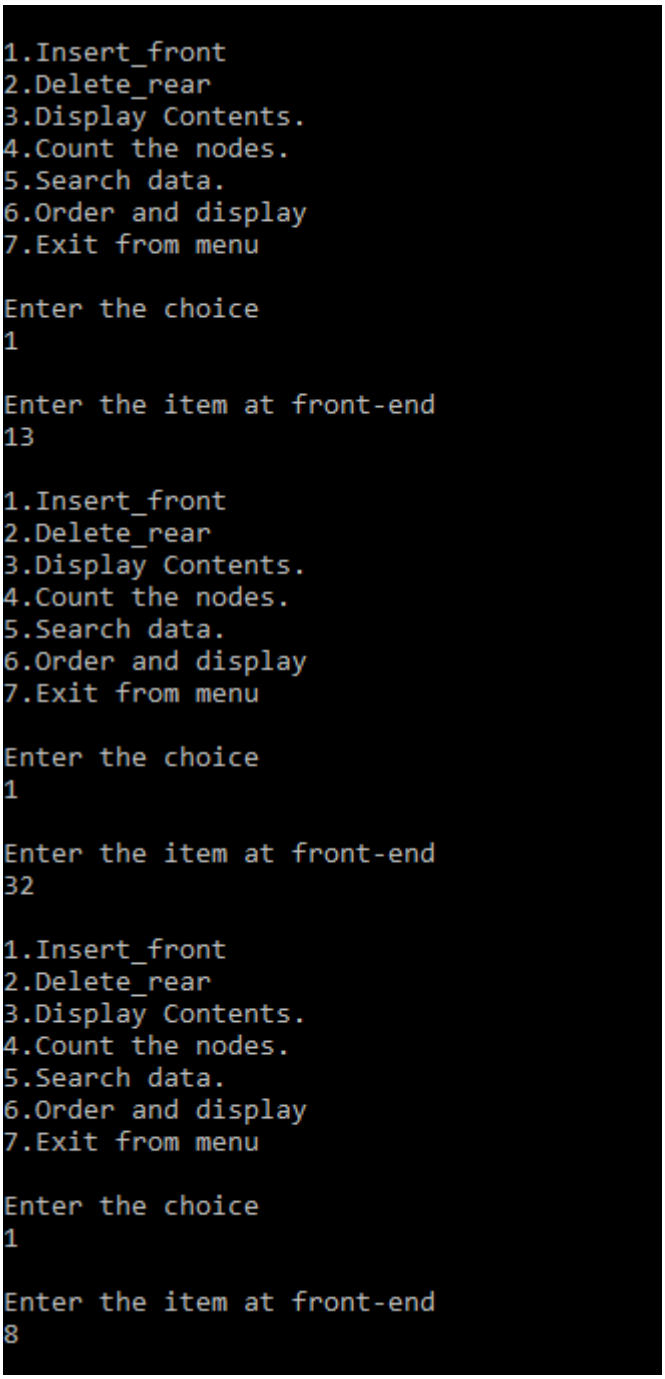
```
}

void search(NODE first)
{
        int item,count=1;
        NODE temp=first;
        printf("Enter  item to be searched\n");
        scanf("%d",&item);
        while(temp->info!=item)
        {
                temp=temp->link;
                count+=1;
        }
        if(temp!=NULL)
        {
                printf("Data is present at %d",count);
        }
        else
        printf("Data is not present");
}

NODE order_list(NODE first)
{
   int swapped, i;
   NODE ptr1,lptr=NULL;

    if (first == NULL)
    return first;
```

```
    do
    {
        swapped = 0;
        ptr1 = first;

        while (ptr1->link != lptr)
        {
            if (ptr1->info > ptr1->link->info)
            {
              int temp = ptr1->info;
              ptr1->info = ptr1->link->info;
              ptr1->link->info = temp;
              swapped = 1;
            }
            ptr1 = ptr1->link;
        }
        lptr = ptr1;
    }
    while (swapped);
    display(first);
}

int main()
{
int item,choice,pos,i,n;
NODE a,b;
NODE first=NULL;

for(;;)
```

```
{
printf("\n1.Insert_front\n2.Delete_rear\n3.Display Contents.\n4.Count the nodes.\n5.Search data.\n6.Order and display\n7.Exit from menu\n\n");
printf("Enter the choice\n");
scanf("%d",&choice);
switch(choice)
{
        case 1:
         printf("\nEnter the item at front-end\n");
         scanf("%d",&item);
         first=insert_front(first,item);
         break;
         case 2:
         first=delete_rear(first);
         break;
         case 3:
         display(first);
         break;
         case 4:
         count(first);
         break;
         case 5:
         search(first);
         break;
         case 6:
         first=order_list(first);
         break;
         case 7:
         exit(0);
         default:
```

```
                printf("Invalid input\n\n");

                break;


}

}

}
```

OUTPUT:



```
1.Insert_front
2.Delete_rear
3.Display Contents.
4.Count the nodes.
5.Search data.
6.Order and display
7.Exit from menu

Enter the choice
5
Enter  item to be searched
32
Data is present at 2
1.Insert_front
2.Delete_rear
3.Display Contents.
4.Count the nodes.
5.Search data.
6.Order and display
7.Exit from menu

Enter the choice
6
8
13
32

1.Insert_front
2.Delete_rear
3.Display Contents.
4.Count the nodes.
5.Search data.
6.Order and display
7.Exit from menu

Enter the choice
4
The length of linked list is: 3
```