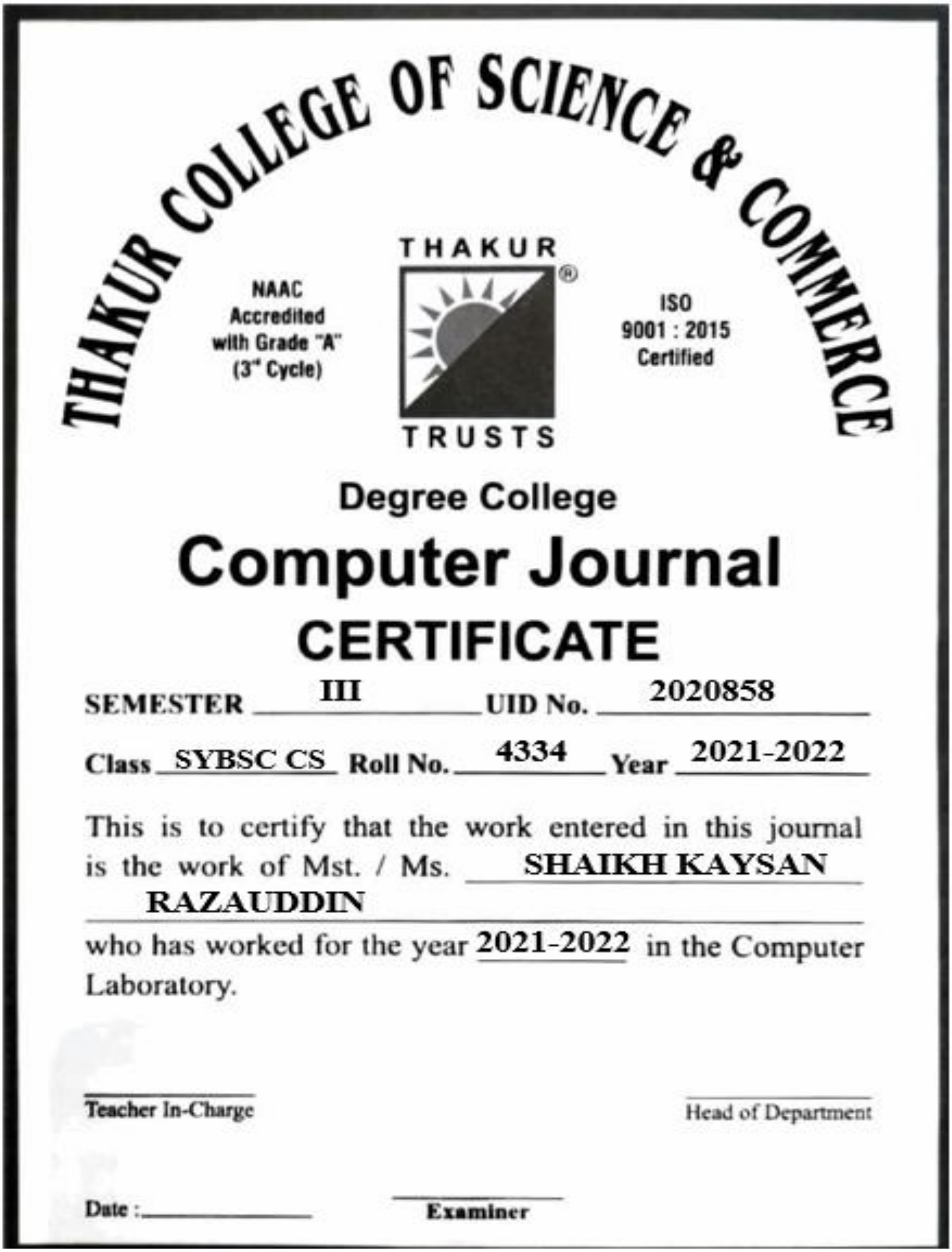


THAKUR COLLEGE OF SCIENCE & COMMERCE

NAAC Accredited with Grade "A" (3rd Cycle)

THAKUR TRUSTS

ISO 9001 : 2015 Certified

Degree College

Computer Journal

CERTIFICATE

SEMESTER III UID No. 2020858

Class SYBSC CS Roll No. 4334 Year 2021-2022

This is to certify that the work entered in this journal is the work of Mst. / Ms. SHAIKH KAYSAN RAZAUDDIN

who has worked for the year 2021-2022 in the Computer Laboratory.

Teacher In-Charge

Head of Department

Date :

Examiner

INDEX

Sr. No.	Practical Name	Page.No.
1.	Implement Linear Search to find an item in a list	**3**
2.	Implement binary search to find an item in an ordered list	**4**
3.	Implementing the Bubble sort sorting algorithm	**5**
4.	Implementing Selection sort sorting algorithm	**6**
5.	Implementing Insertion sort sorting algorithm	**7**
6.	Implementing Merge sort sorting algorithm	**8**
7.	Implement working of Stacks. (pop method to take the last item added off the stack and a push method to add an item to the stack)	**10**
8.	Implement Program for Postfix Evaluation	**12**
9.	Implement the following a) A queue as a list which you add and delete items from. b) A circular queue. (the beginning items of the queue can be reused)	**14**
10.	Implement Linked list and demonstrate the functionality to add and delete items in the linked list.	**16**

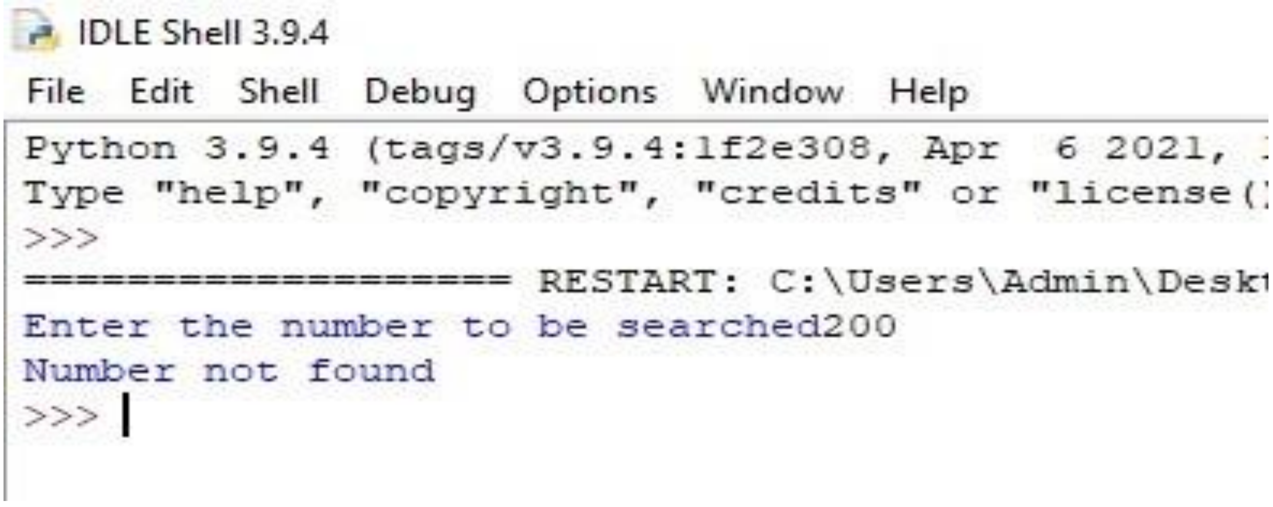
Practical: 01

Aim: Implement Linear Search to find an item in a list.

Source Code:

```
theValues=[105,90,75,56,82,97]
target=int(input("Enter the number to be searched\n"))
def linearSearch(theValues,target):
    n=len(theValues)
    for i in range(n):
        if theValues[i]==target:
            return True
    return False
k=linearSearch(theValues,target)
if k==True:
    print("Number found in the list")
else:
    print("Number not found")
```

Output:



```
Python 3.9.4 (tags/v3.9.4:1f2e308, Apr  6 2021,
Type "help", "copyright", "credits" or "license(
>>>
==================== RESTART: C:\Users\Admin\Deskt
Enter the number to be searched200
Number not found
>>> |
```

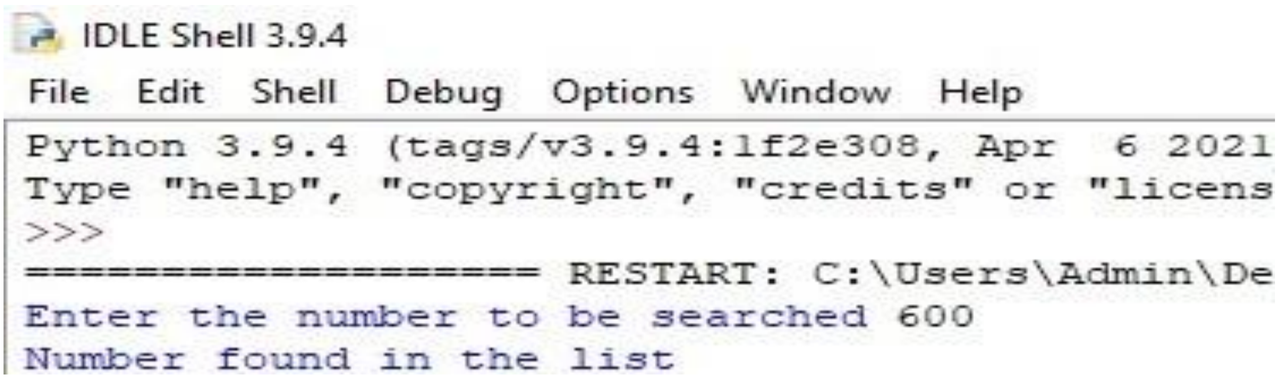
Practical: 02

Aim: Implement binary search to find an item in an ordered list.

Source Code:

```
theValues=[100,200,300,400,500,600,700]
target = int(input("Enter the number to be searched\n"))
def binarySearch(theValues,target):
    low=0
    high=len(theValues)-1
    while low <= high:
        mid=(high+low)//2
        if theValues[mid]==target:
            return True
        elif target < theValues[mid]:
            high = mid-1
        else:
            low = mid+1
    return False
k=binarySearch (theValues,target)
if (k==True):
    print("Number found in the list")
else:
    print("Number not found")
```

Output:



```
Python 3.9.4 (tags/v3.9.4:1f2e308, Apr  6 2021
Type "help", "copyright", "credits" or "licens
>>>
==================== RESTART: C:\Users\Admin\De
Enter the number to be searched 600
Number found in the list
```

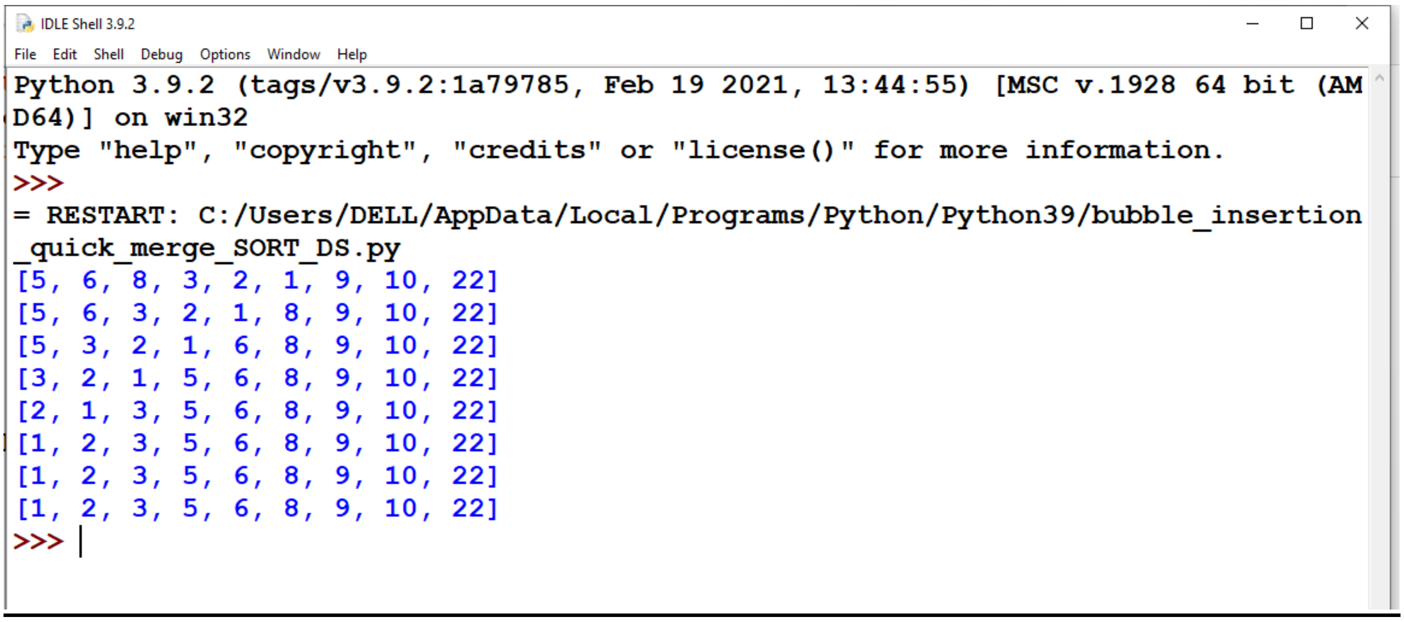
Practical: 03

Aim: Implementing the Bubble sort sorting algorithm

Source Code:

```
theSeq=[5,8,6,9,3,2,1,10,22]
def bubbleSort(theSeq):
    n=len(theSeq)
    for i in range(n-1):
        for j in range(n-1-i):
            if theSeq[j]>theSeq[j+1]:
                tmp=theSeq[j]
                theSeq[j]=theSeq[j+1]
                theSeq[j+1]=tmp
        print(theSeq)
bubbleSort(theSeq)
```

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/DELL/AppData/Local/Programs/Python/Python39/bubble_insertion
_quick_merge_SORT_DS.py
[5, 6, 8, 3, 2, 1, 9, 10, 22]
[5, 6, 3, 2, 1, 8, 9, 10, 22]
[5, 3, 2, 1, 6, 8, 9, 10, 22]
[3, 2, 1, 5, 6, 8, 9, 10, 22]
[2, 1, 3, 5, 6, 8, 9, 10, 22]
[1, 2, 3, 5, 6, 8, 9, 10, 22]
[1, 2, 3, 5, 6, 8, 9, 10, 22]
[1, 2, 3, 5, 6, 8, 9, 10, 22]
>>> |
```

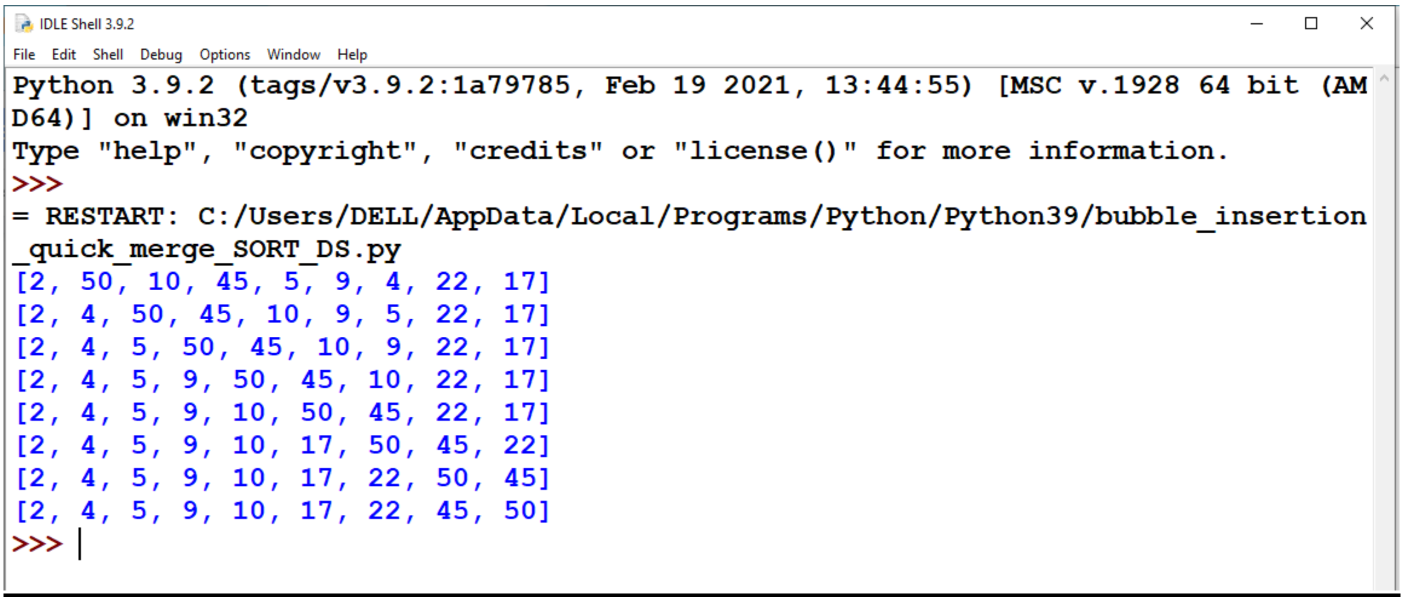
Practical: 04

Aim: Implementing Selection sort sorting algorithm

Source Code:

```
theSeq=[5,50,10,45,2,9,4,22,17]
n=len(theSeq)
def selectionSort(theSeq):
    for i in range(n-1):
        for j in range (i+1,n):
            if theSeq[i]>theSeq[j]:
                tmp=theSeq[i]
                theSeq[i]=theSeq[j]
                theSeq[j]=tmp
        print(theSeq)
k=selectionSort(theSeq)
```

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/DELL/AppData/Local/Programs/Python/Python39/bubble_insertion
_quick_merge_SORT_DS.py
[2, 50, 10, 45, 5, 9, 4, 22, 17]
[2, 4, 50, 45, 10, 9, 5, 22, 17]
[2, 4, 5, 50, 45, 10, 9, 22, 17]
[2, 4, 5, 9, 50, 45, 10, 22, 17]
[2, 4, 5, 9, 10, 50, 45, 22, 17]
[2, 4, 5, 9, 10, 17, 50, 45, 22]
[2, 4, 5, 9, 10, 17, 22, 50, 45]
[2, 4, 5, 9, 10, 17, 22, 45, 50]
>>> |
```

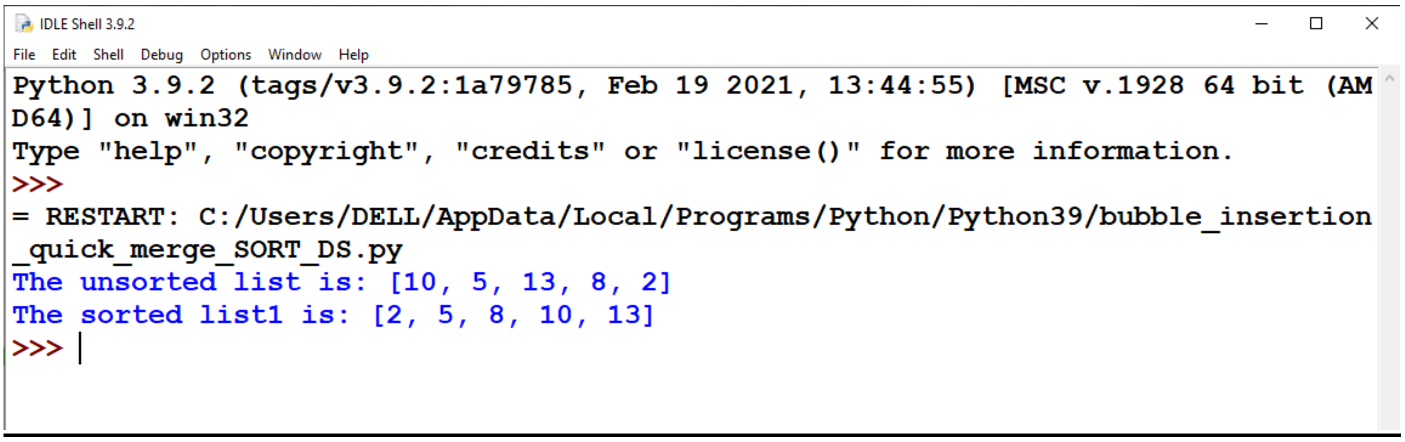
Practical: 05

Aim: Implementing Insertion sort sorting algorithm

Source Code:

```
def insertion_sort(list1):
        for i in range(1, len(list1)):
          value = list1[i]
          j = i - 1
          while j >= 0 and value < list1[j]:
            list1[j + 1] = list1[j]
            j =j- 1
            list1[j + 1] = value
        return list1


list1 = [10, 5, 13, 8, 2]
print("The unsorted list is:", list1)


print("The sorted list1 is:", insertion_sort(list1))
```

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/DELL/AppData/Local/Programs/Python/Python39/bubble_insertion
_quick_merge_SORT_DS.py
The unsorted list is: [10, 5, 13, 8, 2]
The sorted list1 is: [2, 5, 8, 10, 13]
>>> |
```

Practical: 06

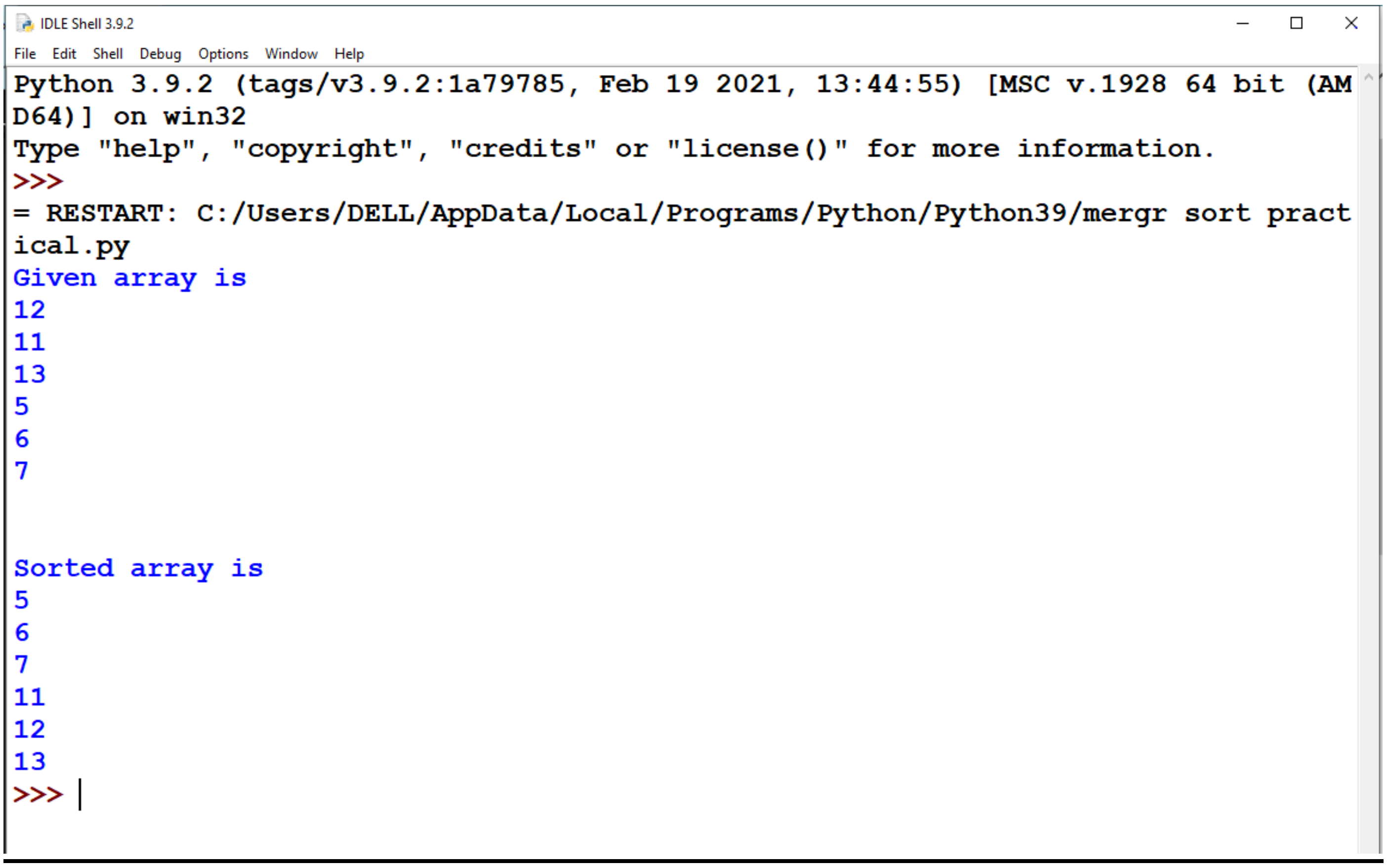
Aim: Implementing Merge sort sorting algorithm

Source Code:

```
def merge(arr, l, m, r):
    n1 = m - l + 1
    n2 = r- m

    L = [0] * (n1)
    R = [0] * (n2)

    for i in range(0 , n1):
        L[i] = arr[l + i]

    for j in range(0 , n2):
        R[j] = arr[m + 1 + j]

    i = 0
    j = 0
    k = l

    while i < n1 and j < n2 :
        if L[i] <= R[j]:
            arr[k] = L[i]
            i += 1
        else:
            arr[k] = R[j]
            j += 1
        k += 1

    while i < n1:
        arr[k] = L[i]
        i += 1
        k += 1

    while j < n2:
        arr[k] = R[j]
        j += 1
        k += 1

def mergeSort(arr,l,r):
    if l < r:

        m = (l+(r-1))//2

        mergeSort(arr, l, m)
        mergeSort(arr, m+1, r)
        merge(arr, l, m, r)
```

```
arr = [12, 11, 13, 5, 6, 7]
n = len(arr)
print ("Given array is")
for i in range(n):
    print ("%d" %arr[i]),

mergeSort(arr,0,n-1)
print ("\n\nSorted array is")
for i in range(n):
    print ("%d" %arr[i]),
```

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/DELL/AppData/Local/Programs/Python/Python39/mergr sort pract
ical.py
Given array is
12
11
13
5
6
7


Sorted array is
5
6
7
11
12
13
>>> |
```

Practical: 07

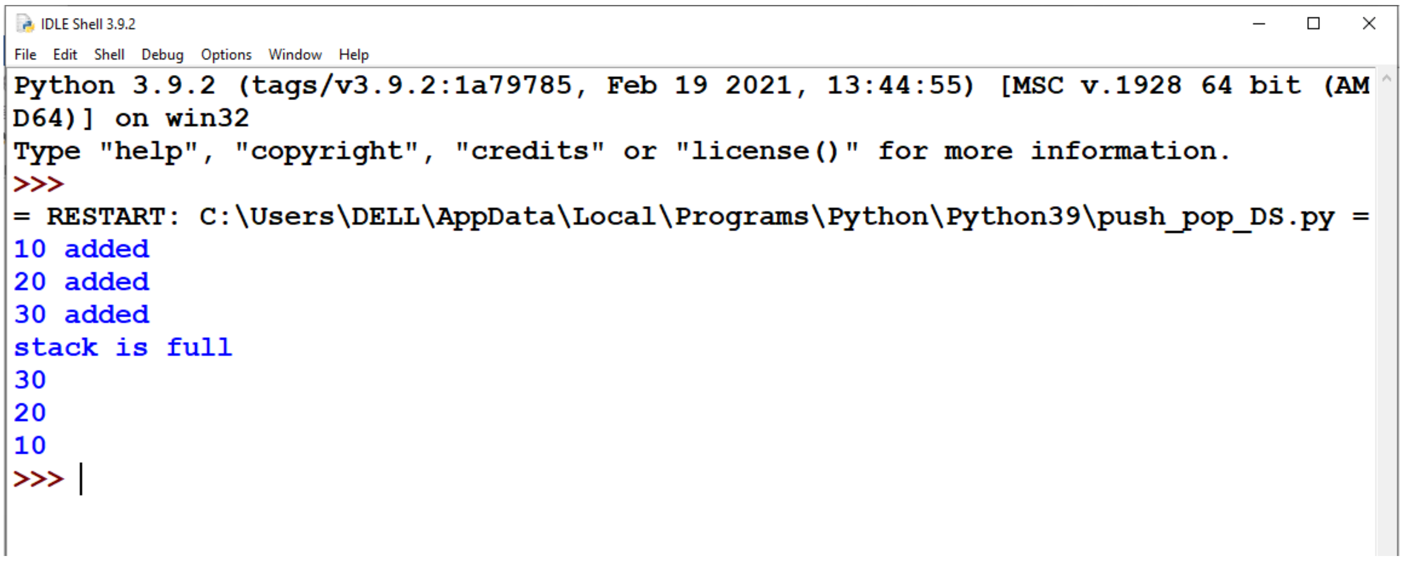
Aim: Implement working of Stacks. (pop method to take the last item added off the stack and a push method to add an item to the stack)

Source Code:

```
class stack:
    s = [2,2,2]
    tos=-1
    def __init__(self):
        self.tos=-1
    def push(self,data):
        if self.tos==2:
            print("stack is full")
        else:
            self.tos+=1
            self.s[self.tos]=data
            print(data,"added")
    def pop(self):
        if self.tos==-1:
            print("stack is empty")
        else:
            print(self.s[self.tos])
            self.tos-=1
s1=stack()
s1.push(10)
s1.push(20)
s1.push(30)
s1.push(40)
s1.pop()
s1.pop()
```

s1.pop()

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\DELL\AppData\Local\Programs\Python\Python39\push_pop_DS.py =
10 added
20 added
30 added
stack is full
30
20
10
>>> |
```

Practical: 08

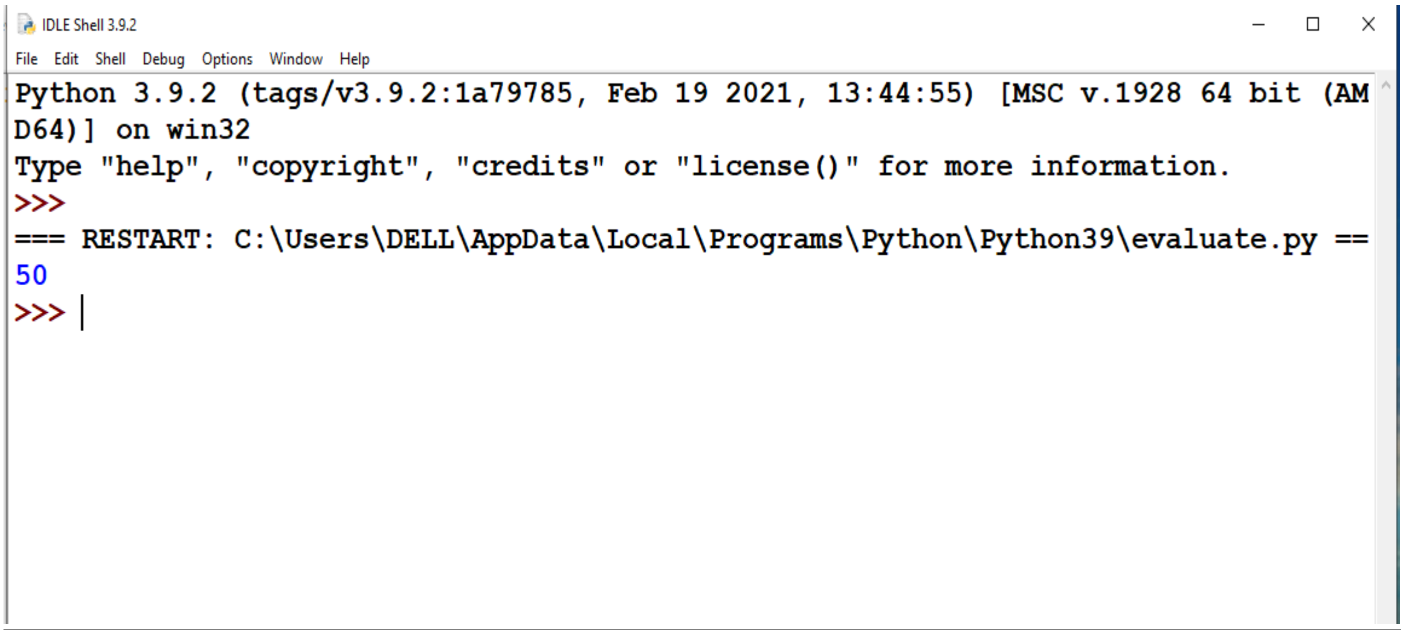
Aim: Implement Program for Postfix Evaluation

Source Code:

```
def evaluate(s):
    k=s.split()
    n=len(k)
    stack=[]
    for i in range(n):
        if k[i].isdigit():
            stack.append(int(k[i]))
        elif k[i]=='+':
            a=stack.pop()
            b=stack.pop()
            stack.append(int(b)+int(a))
        elif k[i]=='-':
            a=stack.pop()
            b=stack.pop()
            stack.append(int(b)-int(a))
        elif k[i]=='*':
            a=stack.pop()
            b=stack.pop()
            stack.append(int(b)*int(a))
        else:
            a=stack.pop()
            b=stack.pop()
            stack.append(int(b)/int(a))
    return stack.pop()
s="8 7 6 * +"
r=evaluate(s)
```

```
print(r)
```

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:\Users\DELL\AppData\Local\Programs\Python\Python39\evaluate.py ==
50
>>>
```

Practical: 09

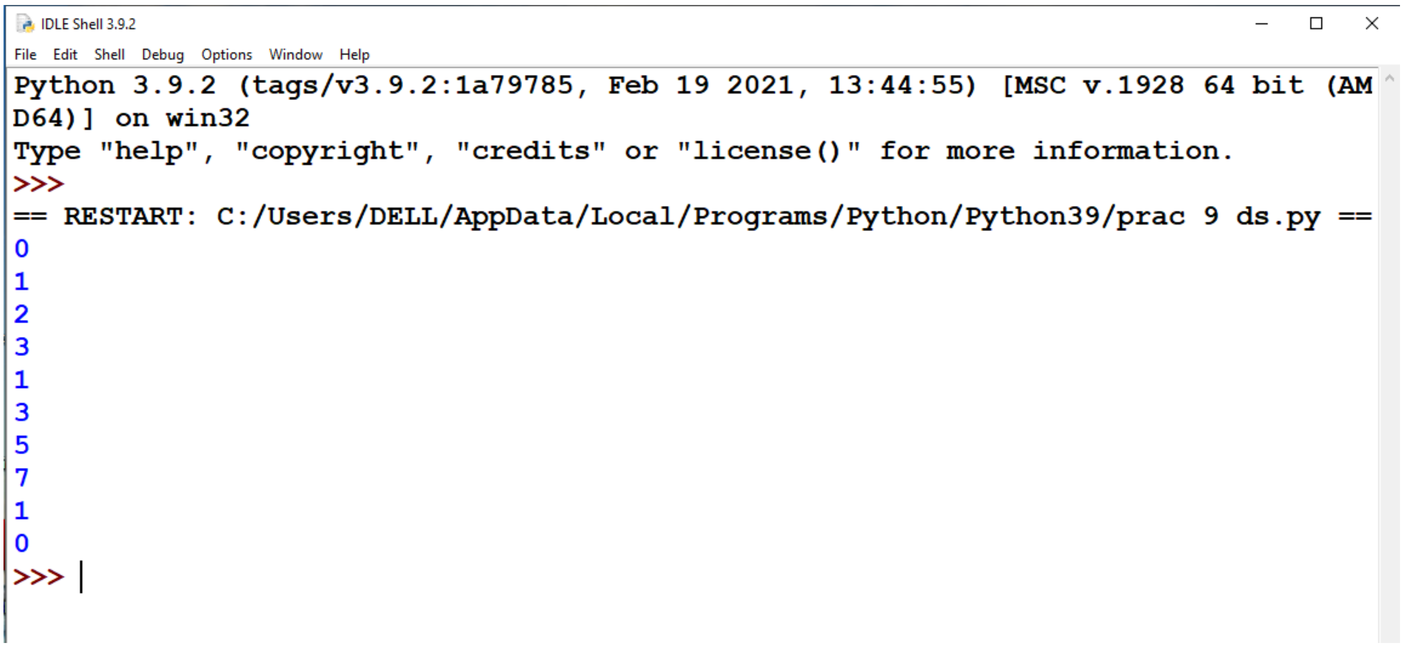
Aim: Implement the following a) A queue as a list which you add and delete items from. b) A circular queue. (the beginning items of the queue can be reused).

Source code:

```
class que:
    s= [-1,-1,-1,-1]
    r=0
    f=0
    def __init__(self):
        self.r=0
        self.f=0

    def enqueue(self,data):
        if self.r==len(self.s) and self.r<=self.f:
            print("queue is full")
        else:
            self.s[self.r]=data
            print(self.r)
            self.r=self.r+1


        if self.r==len(self.s):
            self.r=0
    def dequeue(self):
        if self.f==len(self.s)and self.f<=self.r:
            print("Queue is empty")
        else:
            print(self.s[self.f])
            self.f=self.f+1
        if self.f==len(self.s):
            self.f=0
```

```
q=que()
q.enqueue(1)
q.enqueue(3)
q.enqueue(5)
q.enqueue(7)

q.dequeue()
q.dequeue()
q.dequeue()
q.dequeue()
q.dequeue()
q.enqueue(9)
q.dequeue
```

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:/Users/DELL/AppData/Local/Programs/Python/Python39/prac 9 ds.py ==
0
1
2
3
1
3
5
7
1
0
>>> |
```

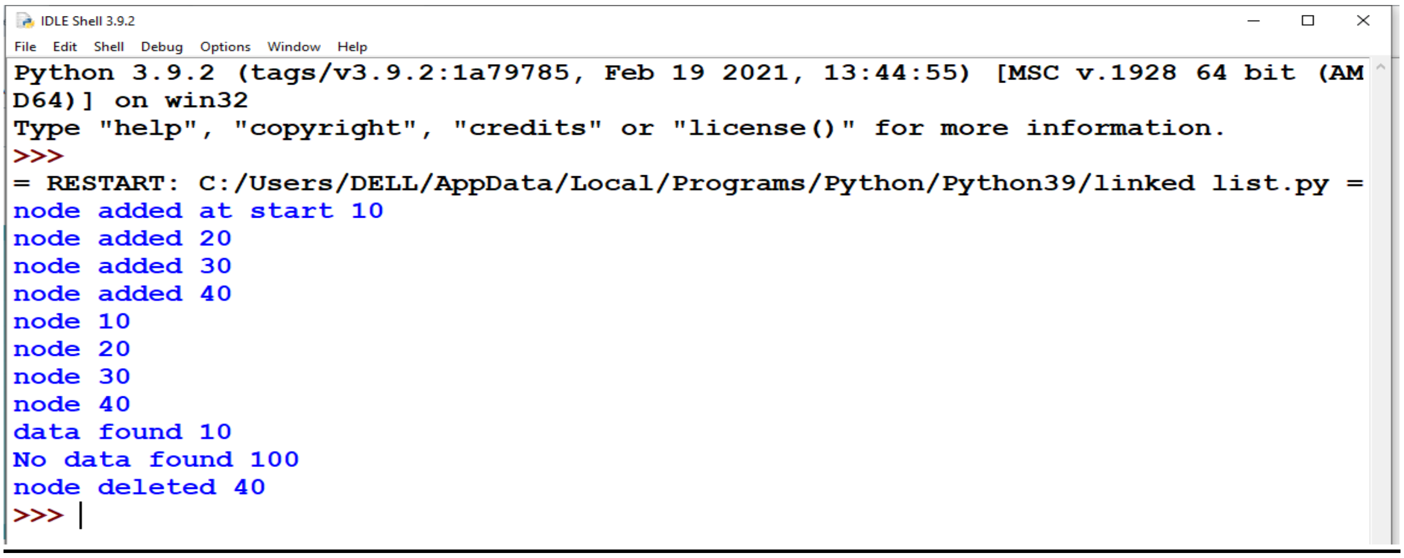
Practical :10

Aim: Implement Linked list and demonstrate the functionality to add and delete items in the linked list.

Source Code:

```
class Node:
    global data
    global next

    def __init__(self,d):
        self.data=d
        self.next=None

class LinkedList:
    global s

    def __init__(self):
        self.s=None

    def add(self,d):
        n=Node(d)
        if self.s==None:
            self.s=n
            print("node added at start",d)
        else:
            h=self.s
            while True:
                if h.next==None:
                    h.next=n
                    print("node added",d)
                    break
                else:
                    h=h.next
    def view(self):
        h=self.s
        while True:
            if h.next!=None:
                print("node",h.data)
                h=h.next
            else:
                print("node",h.data)
                break
    def addbeg(self, d):
        n=Node(d)
        if self.s==None:
            self.s=n
            print("node added at start",d)
        else:
            h=self.s
            self.s=n
            n.next=h
    def search(self,d):
        h=self.s
        while True:
            if h.next!=None:
                if h.data==d:
                    print("data found",d)
                    break
                else:
                    h=h.next
            else:
                print("No data found",d)
                break
    def delete(self,d):
        h=self.s
        if h.next!=None:
            if h.data==d:
                self.s=h.next
                h.next=None
                print("deleted",d)

            else:
                ph=h
                h=h.next
                while True:
                    if h.next!=None or h.data==d:
                        if h.data==d:
                            ph.next=h.next
                            h.next=None
                            print("node deleted",d)
                            break
                        else:
                            ph=h
                            h=h.next
                    else:

                        print("No data found",d)
                        break

ll=LinkedList()
ll.add(10)
ll.add(20)
ll.add(30)
ll.add(40)
ll.view()
ll.search(10)
ll.search(100)
ll.delete(40)
```

Output:



```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bit (AM
D64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/DELL/AppData/Local/Programs/Python/Python39/linked list.py =
node added at start 10
node added 20
node added 30
node added 40
node 10
node 20
node 30
node 40
data found 10
No data found 100
node deleted 40
>>> |
```