**ASSIGNMENT 4**

**Q1. Code**

class Node:

def \_\_init\_\_(self, data):

self.data = data

self.next = None

def push(head, data):

if not head:

return Node(data)

temp = Node(data)

temp.next = head

head = temp

return head

def deleteEnd(head):

if head == None:

return None

if head.next == None:

head = None

return None

second\_last = head

while(second\_last.next.next):

second\_last = second\_last.next

second\_last.next = None

return head

def display():

current = head

while current is not None:

print(current.data, end = ' ')

current = current.next

if \_\_name\_\_=='\_\_main\_\_':

head = None

head = push(head, 1)

head = push(head, 9)

head = push(head, 11)

display ()

print("\n")

head = deleteEnd(head)

display ()

**Q2. Code**

def binary\_search(arr, l, h, x):

if h >= l:

mid = (h + l) // 2

if arr[mid] == x:

return mid

elif arr[mid] > x:

return binary\_search(arr, l, mid - 1, x)

else:

return binary\_search(arr, mid + 1, h, x)

else:

return -1

**Q3. Code**

class Node(object):

def \_\_init\_\_(self, data=None, next=None):

self.data = data

self.next = next

def \_\_str\_\_(self):

return str(self.data)

def print\_nodes(node):

while node:

print (node)

node = node.next

def find\_middle(node):

while node:

current = node

node = node.next

second\_pointer = node.next

next\_pointer = second\_pointer.next

if next\_pointer is None:

return "Middle node is %s" % str(current)

node1 = Node(1)

node2 = Node(2)

node3 = Node(3)

node4 = Node(4)

node1.next = node2

node2.next = node3

node3.next = node4

print (find\_middle(node1))