WEEK – 4 Date: 04/11/2021

1. **Aim**: Develop an application to implement Singly linked list with following operations
2. Insertion
3. Deletion
4. Display

**Code**:

class Node:

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

self.data = data

self.next = None

class LinkedList:

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

self.head = None

def insert(self, new\_data):

new\_node = Node(new\_data)

new\_node.next = self.head

self.head = new\_node

def insertAfter(self, pos, new\_data):

if self.head is None:

return

temp = self.head

if pos == 0:

new\_node = Node(new\_data)

new\_node.next = self.head

self.head = new\_node

return

for i in range(pos - 1):

temp = temp.next

if temp is None:

break

new\_node = Node(new\_data)

new\_node.next = temp.next

temp.next = new\_node

def insertAtEnd(self, new\_data):

new\_node = Node(new\_data)

if self.head is None:

self.head = new\_node

return

last = self.head

while (last.next):

last = last.next

last.next = new\_node

def display(self):

temp = self.head

while (temp):

print (temp.data,end=" ")

temp = temp.next

def deleteNode(self, position):

if self.head is None:

return

temp = self.head

if position == 0:

self.head = temp.next

temp = None

return

for i in range(position - 1):

temp = temp.next

if temp is None:

break

if temp is None:

return

if temp.next is None:

return

next = temp.next.next

temp.next = None

temp.next = next

llist = LinkedList()

print("\*\*\*\*\*\*MENU\*\*\*\*\*\*")

print("\n1.insertion")

print("\n2.display")

print("\n3.delete")

print("\n4.exit")

while True:

n=int(input("\nSelect an option:"))

if n==1:

a = int(input("Enter a number:"))

c = int(input("Select the mode of insertion\n1.At start\n2.At End\n3.At position\nEnter your choice: "))

if c == 1:

llist.insert(a)

if c == 2:

llist.insertAtEnd(a)

if c == 3:

pos = int(input("Enter the position of element you want to Insert: "))

llist.insertAfter(pos, a)

if n==2:

llist.display()

if n==3:

b = int(input("Enter the position of element you want to delete: "))

print("\nAfter deleting an element:")

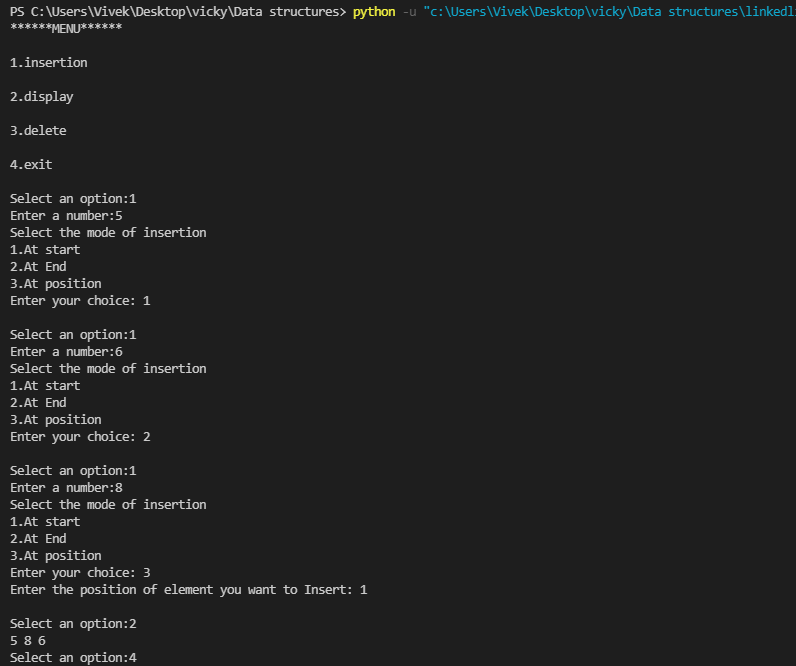
llist.deleteNode(b)

llist.display()

if n==4:

exit(0)

**Output**:



1. **Aim**: Develop an application to implement doubly linked list with following operations
2. insertion
3. Deletion
4. Display(forward and backward)

**Code**:

class Node:

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

self.data = data

self.next = None

self.prev = None

class LinkedList:

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

self.head = None

def insert(self, new\_data):

new\_node = Node(new\_data)

if self.head == None :

new\_node.prev = None

new\_node.next = None

else:

new\_node.prev = None

new\_node.next = self.head

self.head.prev = new\_node

self.head = new\_node

def insertAfter(self, pos, new\_data):

new\_node = Node(new\_data)

if self.head == None :

new\_node.prev = None

new\_node.next = None

self.head = new\_node

if pos == 0:

new\_node.prev = None

new\_node.next = self.head

self.head.prev = new\_node

self.head = new\_node

return

temp = self.head

for i in range(pos - 1):

temp = temp.next

if temp is None:

break

new\_node.next = temp.next

(temp.next).prev = new\_node

temp.next = new\_node

new\_node.prev = temp

def insertAtEnd(self, new\_data):

new\_node = Node(new\_data)

if self.head is None:

new\_node.prev = None

new\_node.next = None

self.head = new\_node

ptr = self.head

while (ptr.next != None):

ptr = ptr.next

ptr.next = new\_node

new\_node.prev = ptr

new\_node.next = None

def display(self):

temp = self.head

print("In forward direction\n")

while (temp.next!=None):

print (temp.data,end=" ")

temp = temp.next

print(temp.data)

print("\nIn backward direction\n")

while(temp.prev!=None):

print (temp.data,end=" ")

temp = temp.prev

print(temp.data)

def delnodefirst(self):

if self.head is None:

print("List Empty!!\n")

else:

temp = self.head

self.head = self.head.next

if(self.head!=None):

self.head.prev = None

def deleteNode(self, position):

if self.head is None:

print("List Empty!!\n")

temp = self.head

if position == 0:

temp = self.head

self.head = self.head.next

if(self.head!=None):

self.head.prev = None

for i in range(position - 1):

temp = temp.next

if temp is None:

break

temp.prev.next = temp.next

temp.next.prev = temp.prev

def delnodelast(self):

if self.head is None:

print("List Empty!!\n")

else:

temp = self.head

while(temp.next != None):

temp = temp.next

temp.prev.next = None

def count(self):

temp = self.head

num = 0

while(temp != None):

num +=1

temp = temp.next

print("\nThe number of elements in list are",num)

llist = LinkedList()

print("\*\*\*\*\*\*MENU\*\*\*\*\*\*")

print("\n1.insertion")

print("\n2.display")

print("\n3.delete")

print("\n4.Count")

print("\n5.exit")

while True:

n=int(input("\nSelect an Option:"))

if n==1:

a = int(input("Enter a number:"))

c = int(input("Select the mode of insertion\n1.At start\n2.At End\n3.At position\nEnter your choice: "))

if c==1:

llist.insert(a)

elif c==2:

llist.insertAtEnd(a)

elif c==3:

pos = int(input("Enter the position: "))

llist.insertAfter(pos,a)

if n==2:

llist.display()

if n==3:

d = int(input("Select the mode of Deletion\n1.At start\n2.At End\n3.At position\nEnter your choice: "))

if(d == 1):

llist.delnodefirst()

elif(d==2):

llist.delnodelast()

elif(d==3):

b = int(input("Enter the position of element you want to delete: "))

llist.deleteNode(b)

print("\nAfter deleting an element:")

llist.display()

if n==4:

llist.count()

if n==5:

exit(0)

**Output**:

