class Node:

def \_\_init\_\_(self, e, n, p):

self.element = e

self.next = n

self.prev = p

class DoublyList:

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

if type(a)==list:

self.head=Node(a[0],None,None)

tail=self.head

for i in range(1,len(a)):

n=Node(a[i],None,None)

tail.next=n

n.prev=tail

tail=n

else:

self.head=a

def countNode(self):

temp=self.head

c=0

while temp!=None:

temp=temp.next

c+=1

return c

def forwardprint(self):

temp=self.head

count= self.countNode()

c=0

while temp!=None:

if c!=count-1:

print(temp.element,end=", ")

else:

print(temp.element)

print()

temp=temp.next

c+=1

def backwardprint(self):

count=self.countNode()

temp=self.head

c=0

while c!=count-1:

temp=temp.next

c+=1

x=count-1

while temp!=None:

if x!=0:

print(temp.element,end=", ")

else:

print(temp.element)

print()

x-=1

temp=temp.prev

def nodeAt(self, idx):

count=self.countNode()

temp=self.head

c=0

if 0<idx<count:

while c!=idx+1:

if c==idx:

return temp

c+=1

temp=temp.next

else:

return None

def indexOf(self, elem):

temp=self.head

c=0

flag=False

while temp!=None:

if temp.element==elem:

flag=True

break

else:

c+=1

flag=False

temp=temp.next

if flag==True:

return c

else:

return -1

def insert(self, elem, idx):

len=self.countNode()

if idx>len:

return 'Invalid index'

else:

if idx==0:

n = Node (elem, None, None)

n.next= self.head

self.head.prev=n

self.head=n

elif idx==len:

n = Node (elem, None, None)

p=self.nodeAt(idx-1)

n.prev = p

p.next = n

else:

n = Node (elem, None, None)

p=self.nodeAt(idx-1)

q=p.next

p.next=n

q.prev=n

n.prev=p

n.next=q

def remove(self, idx):

if idx==0:

val=self.head.element

self.head=self.head.next

self.head.prev=None

return str(val)

else:

c=0

temp=self.head

length=self.countNode()

if length-1==idx:

temp=self.nodeAt(idx)

val=temp.element

temp.prev.next=None

return str(val)

elif length <= idx:

return("index invalid")

else:

for i in range(idx):

temp=temp.next

val=temp.element

temp.prev.next=temp.next

temp.next.prev=temp.prev

return str(val)

#..............................

print("/// Test 01 ///")

a1 = [10, 20, 30, 40]

h1 = DoublyList(a1) # Creates a linked list using the values from the array

h1.forwardprint() # This should print: 10,20,30,40.

h1.backwardprint() # This should print: 40,30,20,10.

print(h1.countNode()) # This should print: 4

print("/// Test 02 ///")

# returns the reference of the at the given index. For invalid idx return None.

myNode = h1.nodeAt(2)

print(myNode.element) # This should print: 30. In case of invalid index This will print "index error"

print("/// Test 03 ///")

# returns the index of the containing the given element. if the element does not exist in the List, return -1.

index = h1.indexOf(40)

print(index) # This should print: 3. In case of element that

#doesn't exists in the list this will print -1.

print("/// Test 04 ///")

a2 = [10, 20, 30, 40]

h2 = DoublyList(a2) # uses the constructor

h2.forwardprint() # This should print: 10,20,30,40.

# inserts containing the given element at the given index. Check validity of index.

h2.insert(85,0)

h2.forwardprint() # This should print: 85,10,20,30,40.

h2.backwardprint() # This should print: 40,30,20,10,85.

print()

h2.insert(95,3)

h2.forwardprint() # This should print: 85,10,20,95,30,40.

h2.backwardprint() # This should print: 40,30,95,20,10,80.

print()

h2.insert(75,6)

h2.forwardprint() # This should print: 85,10,20,95,30,40,75.

h2.backwardprint() # This should print: 75,40,30,95,20,10,85.

print("/// Test 05 ///")

a3 = [10, 20, 30, 40, 50, 60, 70]

h3 = DoublyList(a3) # uses the constructor

h3.forwardprint() # This should print: 10,20,30,40,50,60,70.

# removes at the given index. returns element of the removed node. Check validity of index. return None if index is invalid.

print("Removed element: "+ h3.remove(0)) # This should print: Removed element: 10

h3.forwardprint() # This should print: 20,30,40,50,60,70.

h3.backwardprint() # This should print: 70,60,50,40,30,20.

print("Removed element: "+ h3.remove(3)) # This should print: Removed element: 50

h3.forwardprint() # This should print: 20,30,40,60,70.

h3.backwardprint() # This should print: 70,60,40,30,20.

print("Removed element: "+ h3.remove(4)) # This should print: Removed element: 70

h3.forwardprint() # This should print: 20,30,40,60.

h3.backwardprint() # This should print: 60,40,30,20.