class Node:

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

self.element = e

self.next = n

self.prev = p

class DoublyList:

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

dummy = Node(None,None,None)

self.head = dummy

currentHead = dummy

for i in range(0,len(a)):

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

currentHead.next = NODE

NODE.prev = currentHead

currentHead = NODE

currentHead.next = self.head

self.head.prev = currentHead

def countNode(self):

currentNode = self.head.next

count = 0

while (currentNode != self.head):

count += 1

currentNode = currentNode.next

return count

def forwardprint(self):

currentNode = self.head.next

while (currentNode.next != self.head):

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

currentNode = currentNode.next

print(currentNode.element)

def backwardprint(self):

currentNode = self.head.prev

while (currentNode.prev != self.head):

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

currentNode = currentNode.prev

print(currentNode.element)

def nodeAt(self, idx):

if idx < 0 or idx >+ self.countNode():

return 'Invalid'

count = 0

currentNode = self.head.next

while (currentNode != self.head):

if count == idx:

return currentNode

count += 1

currentNode = currentNode.next

def indexOf(self, elem):

currentNode = self.head.next

count = 0

while (currentNode != self.head):

if currentNode.element == elem:

return count

count += 1

currentNode = currentNode.next

return 'Invalid'

def insert(self, elem, idx):

if (idx < 0 or idx > self.countNode()):

print("Invalid")

else:

NODE = Node(elem, None, None)

if (idx == 0):

var = self.head.next

NODE.next = var

var.prev = NODE

NODE.prev = self.head

self.head.next = NODE

else:

prev\_node = self.nodeAt(idx-1)

NODE.next = prev\_node.next

prev\_node.next.prev = NODE

prev\_node.next = NODE

NODE.prev = prev\_node

def remove(self, idx):

if idx < 0 or idx >= self.countNode():

return 'Invalid'

remove = self.nodeAt(idx)

pred\_node = remove.prev

next\_node = remove.next

pred\_node.next = next\_node

next\_node.prev = pred\_node

return str(remove.element)

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.