```
class Node:
  def __init__(self, e, n, p):
    self.element = e
    self.next = n
    self.prev = p

class DoublyList:

  def __init__(self, a):
  #  Design the constructor based on data type of a. If 'a' is built in python list then
  #  Creates a linked list using the values from the given array.
    self.head = None
    # To Do
    pass # Remove this line

  # Counts the number of Nodes in the list
  def countNode(self):
    # To Do
    pass # Remove this line

  # prints the elements in the list
  def forwardprint(self):
    # To Do
    pass # Remove this line

  # prints the elements in the list backward
  def backwardprint(self):
    # To Do
    pass # Remove this line

  # returns the reference of the at the given index. For invalid index return None.
  def nodeAt(self, idx):
    # To Do
    pass # Remove this line

  # returns the index of the containing the given element. if the element does not exist in the List,
return -1.
  def indexOf(self, elem):
    # To Do
    pass # Remove this line

  # inserts containing the given element at the given index Check validity of index.
  def insert(self, elem, idx):
```

```
    # To Do
    pass # Remove this line
  # removes at the given index. returns element of the removed node. Check validity of index.
return None if index is invalid.
  def remove(self, idx):
    # To Do
    pass # Remove this line


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.
```