#

# [21] Merge Two Sorted Lists

#

# https://leetcode.com/problems/merge-two-sorted-lists/description/

#

# algorithms

# Easy (43.75%)

# Total Accepted: 433.8K

# Total Submissions: 991.2K

# Testcase Example: '[1,2,4]\n[1,3,4]'

#

# Merge two sorted linked lists and return it as a new list. The new list

# should be made by splicing together the nodes of the first two lists.

#

# Example:

#

# Input: 1->2->4, 1->3->4

# Output: 1->1->2->3->4->4

#

#

#

# Definition for singly-linked list.

# class ListNode:

# def \_\_init\_\_(self, x):

# self.val = x

# self.next = None

class Solution:

def mergeTwoLists(self, l1, l2):

if l1 is None and l2 is None:

#print("1")

return None

if not l1:

#print("2")

return l2

if not l2:

#print("3")

return l1

p=l1

q=l2

# if l1.val<=l2.val:

# p=l1

# q=l2

# else:

# p=l2

# q=l1

while p is not None and q is not None:

pcopy=p

pp=p

global qq

qq=q

qcopy=q

flag =True

# pval=p.val

# qval=q.val

#print(p.val,q.val)

if p is not None and q is not None:

while(p.val<q.val) :

global pp

global p

global q

pp=p

p=p.next

if p is None:

break

if pp.val<q.val:

pp.next=q

if pp.val==q.val:

pp=p

p=p.next

pp.next=q

flag=False

if p is not None and q is not None:

while(q.val<p.val) :

global p

global q

global qq

qq=q

q=q.next

if q is None:

break

if qq.val<p.val:

qq.next=p

if p.val==qq.val :

q=q.next

qq.next=p

if p is None:

p=q

break

if q is None:

q=p

break

if l1.val<=l2.val:

return l1

else: return l2