```
Student Name: Chufeng Jiang
Student No.: 19949
Leetcode Homepage: https://leetcode.com/u/Chufeng_Jiang/

Submission Details:
Q1: https://leetcode.com/submissions/detail/1261602096/
Q2: https://leetcode.com/submissions/detail/1261662299/
Q3: https://leetcode.com/submissions/detail/1261665319/
```

**Module-2 Homework**

- Give the code for each question
- Give me a clear expalanation of your solution way for each question
- Submit the notebook as well as its pdf version
  Question 1: Leetcode Question 189. Easy. "Rotate Array"

Given an integer array nums, rotate the array to the right by k steps, where k is non-negative.

Example 1:

Input: nums = [1,2,3,4,5,6,7], k = 3
Output: [5,6,7,1,2,3,4]
Explanation:
rotate 1 steps to the right: [7,1,2,3,4,5,6]
rotate 2 steps to the right: [6,7,1,2,3,4,5]
rotate 3 steps to the right: [5,6,7,1,2,3,4]
Example 2:

Input: nums = [-1,-100,3,99], k = 2
Output: [3,99,-1,-100]
Explanation:
rotate 1 steps to the right: [99,-1,-100,3]
rotate 2 steps to the right: [3,99,-1,-100]

```
class Solution:
    def rotate(self, nums, k):
        """
        :type nums: List[int]
        :type k: int
        :rtype: None Do not return anything, modify nums in-place instead.
        """

        """we define a reverse function to reverse the elements from index_l to
index_r in the given nums array"""
        def reverse(index_l, index_r):
            while index_l < index_r:
                nums[index_l], nums[index_r] = nums[index_r], nums[index_l]
                index_l += 1
                index_r -= 1

        """ special cases"""
        if not nums or k == 0: return
```

```
        """
        Input: nums = [1,2,3,4,5,6,7], k = 3

        1st, we reverse the whole array --> [7,6,5,4,3,2,1]
        2nd, we slice the array into two parts --> [7,6,5] [4,3,2,1]
        3rd, we reverse the individual two arrays separatly --> [5,6,7] [1,2,3,4]

        finally, we get the result and return the array.
        """
        size = len(nums) # get the length of the array

        """
        To prevent the overflow of k.
        For example when k(103) > size(7),
        the residule of k % size is 5 (103/7=14---5);
        so we rotate the array to the right by 5 steps which is equal to 103
steps.
        """
        k %= size # revise the steps to rotate"""

        reverse(0, size - 1) # [1,2,3,4,5,6,7] --> [7,6,5,4,3,2,1]
        reverse(0, k-1) # [7,6,5] -->[5,6,7]
        reverse(k, size - 1) # [4,3,2,1] --> [1,2,3,4]
        return None

"""""""""""""""""""""""""""""""""""""""
def test():
    solution = Solution()
    nums = [1, 2, 3, 4, 5, 6, 7]
    k = 3
    solution.rotate(nums, k)
    expected_array = [5,6,7,1,2,3,4]
    if nums == expected_array:
        print("Q1 Test case passed.\n")

test()
```

```
Q1 Test case passed.
```

Question 2: Leetcode Question 665. Medium. "Non-decreasing Array"

Given an array nums with n integers, your task is to check if it could become non-decreasing by modifying at most one element.

We define an array is non-decreasing if nums[i] <= nums[i + 1] holds for every i (0-based) such that (0 <= i <= n - 2).

Example 1:

Input: nums = [4,2,3]
Output: true
Explanation: You could modify the first 4 to 1 to get a non-decreasing array.
Example 2:

Input: nums = [4,2,1]
Output: false
Explanation: You cannot get a non-decreasing array by modifying at most one element.

Constraints:

n == nums.length

```
class Solution(object):
    def checkPossibility(self, nums):
        """
        :type nums: List[int]
        :rtype: bool
        """

        """ We can only modify at most one element, that means we can only change
one of the elements for 1 time."""
        flag = True

        """
        We traverse the array,and each time we compare 3 consecutive elements,
        because we have to consider that after the modification, whether the new
array can satisfy
        that ① the visited elements are non-decreasing and ②the incoming elements
are also non-decreasing.

        Input = [3,4,2,5]:
                        5
                  /4     /
              3 /  \\   /
                     \\2
        In this case, we can modify 4 or 2 to make a non-decreasing order.
        However, since the trough 2 is even smaller than the number of 3 which is
2 position in front of it:
        (1) if we reduce 4 to 2, then [3,2,2,|...] doesn't satisfy our
requirement, and need further modification.
        (1) if we increase 2 to 4, then [3,4,4,|...] satisfy our requirement.

        Input = [1,4,2,5]:
                           /5
                    /4     /
                   /  \\   /
                  /    \\2/
              1 /
        In this case,  we only need to modify 4, because the trough of 2 is
higher than the minimum 1 which is 2 position in front of it.

        """
        for i in range(len(nums) - 1):
            if nums[i] > nums[i + 1]:  # [3,4,2,5], i=1, nums[1]= 4 > nums[2]=2
                if flag == False:
```

```
                    return False

                if i == 0 or nums[i + 1] >= nums[i - 1]:  # [1,4,2,5], i = 1,
nums[1]= 4 > nums[2]=2
                    nums[i] = nums[i + 1]  # [1,2,2,5]

                else:  # [3,4,2,5], i=1, nums[2]=2 < nums[0]=3
                    nums[i + 1] = nums[i]  # [3,4,4,5]

                flag = False
        return True


"""""""""""""""""""""""""""""""""""
def test():
    solution = Solution()
    nums = [4, 2, 1]

    a = solution.checkPossibility(nums)

    if a == False:
        print("Q2 Test case passed.\n")


test()
```

```
Q2 Test case passed.
```

```
```

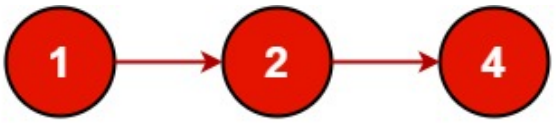
Question 3: Leetcode Question 21. Easy. "Merge Two Sorted List"

ou are given the heads of two sorted linked lists list1 and list2.

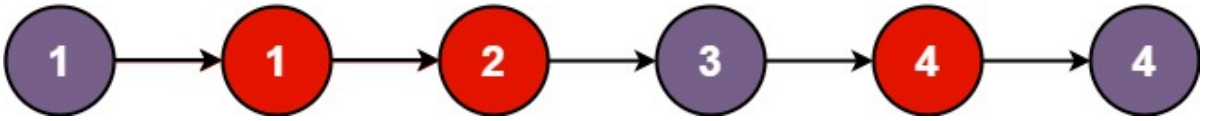
Merge the two lists into one sorted list. The list should be made by splicing together the nodes of the first two lists.

Return the head of the merged linked list.

Example 1:








Input: list1 = [1,2,4], list2 = [1,3,4]
Output: [1,1,2,3,4,4]
Example 2:

Input: list1 = [], list2 = []
Output: []
Example 3:

Input: list1 = [], list2 = [0]
Output: [0]

Constraints:

The number of nodes in both lists is in the range [0, 50].
-100 <= Node.val <= 100
Both list1 and list2 are sorted in non-decreasing order.

```
class ListNode(object):
    def __init__(self, val=0, next=None):
        self.val = val
        self.next = next

class Solution(object):
    def mergeTwoLists(self, list1, list2):
        """
        :type list1: Optional[ListNode]
        :type list2: Optional[ListNode]
        :rtype: Optional[ListNode]
        """

        """
        We can create a dummy node to start our new linkedlist, and declare a
cursor to mark current node pointing to this dummy node.
        """
        cur = dum = ListNode(0)

        """
        Then we compare the values of the pointed nodes in the existed sorted
linked lists:
            (1) we linked the node with smaller value to the next pointer of the
current node;
```

```
            (2) if the values are the same, we firstly linked the node from the
second list.
            (3) once one of the linkedlists end, the next pointer of the current
node will linked to the rest part of the other linkedlist.
        """
        while list1 and list2:
            if list1.val < list2.val: ## case(1)
                cur.next, list1 = list1, list1.next
            else: ## case(2)
                cur.next, list2 = list2, list2.next
            cur = cur.next

        cur.next = list1 if list1 else list2 # case (3)

        return dum.next

"""""""""""""""""""""""""""""""""""
def linked_list_to_list(node):
    result = []
    while node:
        result.append(node.val)
        node = node.next
    return result

def test():
    solution = Solution()
    node1_1 = ListNode(1)
    node1_2 = ListNode(2)
    node1_4 = ListNode(4)
    node1_1.next = node1_2
    node1_2.next = node1_4

    node2_1 = ListNode(1)
    node2_2 = ListNode(3)
    node2_4 = ListNode(4)
    node2_1.next = node2_2
    node2_2.next = node2_4

    q3 = solution.mergeTwoLists(node1_1, node2_1)
    result_list = linked_list_to_list(q3)
    expected_list = [1,1,2,3,4,4]

    if result_list == expected_list:
        print("Q3 Test case passed.\n")

test()
```

```
Q3 Test case passed.
```