def get\_smaller\_to\_right\_count(arr):  
 counts = []  
 for i in range(len(arr)):  
 count = 0  
 for j in range(i, len(arr)):  
 if arr[j] < arr[i]:  
 count = count + 1  
 counts.append(count)  
 return counts

PYTHON

def merge(arr1, arr2, count):  
 arr = []  
 i, j = 0, 0  
 while i < len(arr1) and j < len(arr2): # loop until the end of the shorter array  
 if arr1[i] <= arr2[j]: # item at top of arr1 is smaller  
 arr.append(arr1[i])  
 count[arr1[i]] = count[arr1[i]] + j  
 i = i + 1  
 else: # item at top of arr2 is smaller  
 arr.append(arr2[j])  
 j = j + 1  
  
 if i != len(arr1): # add all remaining items from the longer array  
 arr += arr1[i:]  
 for item in arr1[i:]:  
 count[item] = count[item] + j  
 else:  
 arr += arr2[j:]  
  
 return arr, count  
  
  
def mergesort(arr, count):  
 if len(arr) <= 1:  
 return arr, count  
 mid = len(arr) // 2  
 arr1, count = mergesort(arr[:mid], count)  
 arr2, count = mergesort(arr[mid:], count)  
 return merge(arr1, arr2, count)

def get\_smaller\_to\_right\_count(arr):  
 # creating hash map  
 count = {}  
 for item in arr:  
 count[item] = 0  
  
 \_, count = mergesort(arr, count)  
  
 result = []  
 for item in arr:  
 result.append(count[item])  
  
 return result

PYTHON

def get\_median(arr1, arr2):  
 arr = merge(arr1, arr2)  
 if len(arr) % 2 != 0:  
 return arr[len(arr) // 2]  
 else:  
 mid = len(arr) // 2  
 return (arr[mid - 1] + arr[mid]) / 2

PYTHON

def get\_median\_single(arr):  
 n = len(arr)  
 mid = n // 2  
 if n % 2 == 0:  
 return (arr[mid] + arr[mid - 1]) / 2  
 return arr[mid]  
  
def get\_median(arr1, arr2):  
 len1, len2 = len(arr1), len(arr2)  
 i, j = len1 // 2, len2 // 2  
  
 if len1 == 1 and len2 == 1:  
 return (arr1[0] + arr2[0]) / 2  
  
 m = get\_median\_single(arr1)  
 n = get\_median\_single(arr2)

if m == n:  
 return m # or n  
 if m < n:  
 return get\_median(arr1[i:], arr2[:j])  
 return get\_median(arr1[:i], arr2[j:])

PYTHON