**class** Solution:  
 **def** PartitionOfK(self, numbers, start, end, k):  
 **if** k < 0 **or** numbers == [] **or** start < 0 **or** end >= len(numbers) **or** k > end:  
 **return** low, high = start, end  
 key = numbers[low]  
 **while** low < high:  
 **while** low < high **and** numbers[high] >= key:  
 high -= 1  
 numbers[low] = numbers[high]  
 **while** low < high **and** numbers[low] <= key:1   
 low += 1  
 numbers[high] = numbers[low]  
 numbers[low] = key  
 **if** low < k:  
 self.PartitionOfK(numbers, start + 1, end, k)  
 **elif** low > k:   
 self.PartitionOfK(numbers, start, end - 1, k)  
 **def** GetLeastNumbers\_Solution(self, tinput, k):  
 *# write code here* **if** k <= 0 **or** tinput == [] **or** k > len(tinput):  
 **return** []  
 self.PartitionOfK(tinput, 0, len(tinput) - 1, k)  
 **return** sorted(tinput[0:k])  
*#测试：*sol = Solution()  
listNum = [4,5,1,6,2,7,3,8]  
rel = sol.GetLeastNumbers\_Solution(listNum, 4)  
print(rel)

**class** Solution:  
 **def** GetLeastNumbers\_Solution(self, tinput, k):  
 *# write code here* **if** tinput == [] **or** k <= 0 **or** k > len(tinput):  
 **return** []  
 result = []  
 **for** num **in** tinput:  
 **if** len(result) < k:  
 result.append(num)  
 **else**:  
 **if** num < max(result):  
 result[result.index(max(result))] = num  
 **return** sorted(result)  
*#测试：*sol = Solution()  
listNum = [4,5,1,6,2,7,3,8]  
rel = sol.GetLeastNumbers\_Solution(listNum, 4)  
print(rel)