**WORKSHEET 4**

**Student Name:** ANMOLPREET SINGH **UID:** 22BCS15759

**Branch:** BE-CSE **Section/Group:** 22BCS\_NTPP-602-A

**Semester:** 6th **Date of Performance:** 13/01/2025

**Subject Name:** AP LAB - II **Subject Code:** 22CSP-351

1. **Aim:** Reverse You are given two integer arrays nums1 and nums2, sorted in **non-decreasing order**, and two integers m and n, representing the number of elements

in nums1 and nums2 respectively.

**Merge** nums1 and nums2 into a single array sorted in **non-decreasing order**.

The final sorted array should not be returned by the function, but instead be *stored inside the array* nums1. To accommodate this, nums1 has a length of m + n, where the first m elements denote the elements that should be merged, and the last n elements are set to 0 and should be ignored. nums2 has a length of n.

# Source Code:

class Solution:

def merge(self, nums1: List[int], m: int, nums2: List[int], n: int) -> None:

"""

Do not return anything, modify nums1 in-place instead.

"""

p1 = m - 1

p2 = n - 1

p = m + n - 1

while p1 >= 0 and p2 >= 0:

if nums1[p1] > nums2[p2]:

nums1[p] = nums1[p1]

p1 -= 1

else:

nums1[p] = nums2[p2]

p2 -= 1

p -= 1

while p2 >= 0:

nums1[p] = nums2[p2]

p2 -= 1

p -= 1

# Screenshots of outputs:

**2.**

**Aim:** Given an integer array nums and an integer k, return *the* k *most frequent elements*. You may return the answer in **any order**.

# Source Code:

import heapq

from collections import Counter

class Solution:

def topKFrequent(self, nums: List[int], k: int) -> List[int]:

"""

Given an integer array nums and an integer k, return the k most frequent elements.

You may return the answer in any order.

"""

count = Counter(nums)

return heapq.nlargest(k, count.keys(), key=count.get)

# Screenshots of outputs:

**3.**

**Aim:** Given an array of intervals where intervals[i] = [starti, endi], merge all overlapping intervals, and return *an array of the non-overlapping intervals that cover all the intervals in the input*.

# Source Code:

class Solution:

def merge(self, intervals: List[List[int]]) -> List[List[int]]:

"""

Given an array of intervals where intervals[i] = [starti, endi], merge all overlapping intervals,

and return an array of the non-overlapping intervals that cover all the intervals in the input.

"""

intervals.sort(key=lambda x: x[0]) # Sort by start times

merged = []

for interval in intervals:

if not merged or merged[-1][1] < interval[0]:

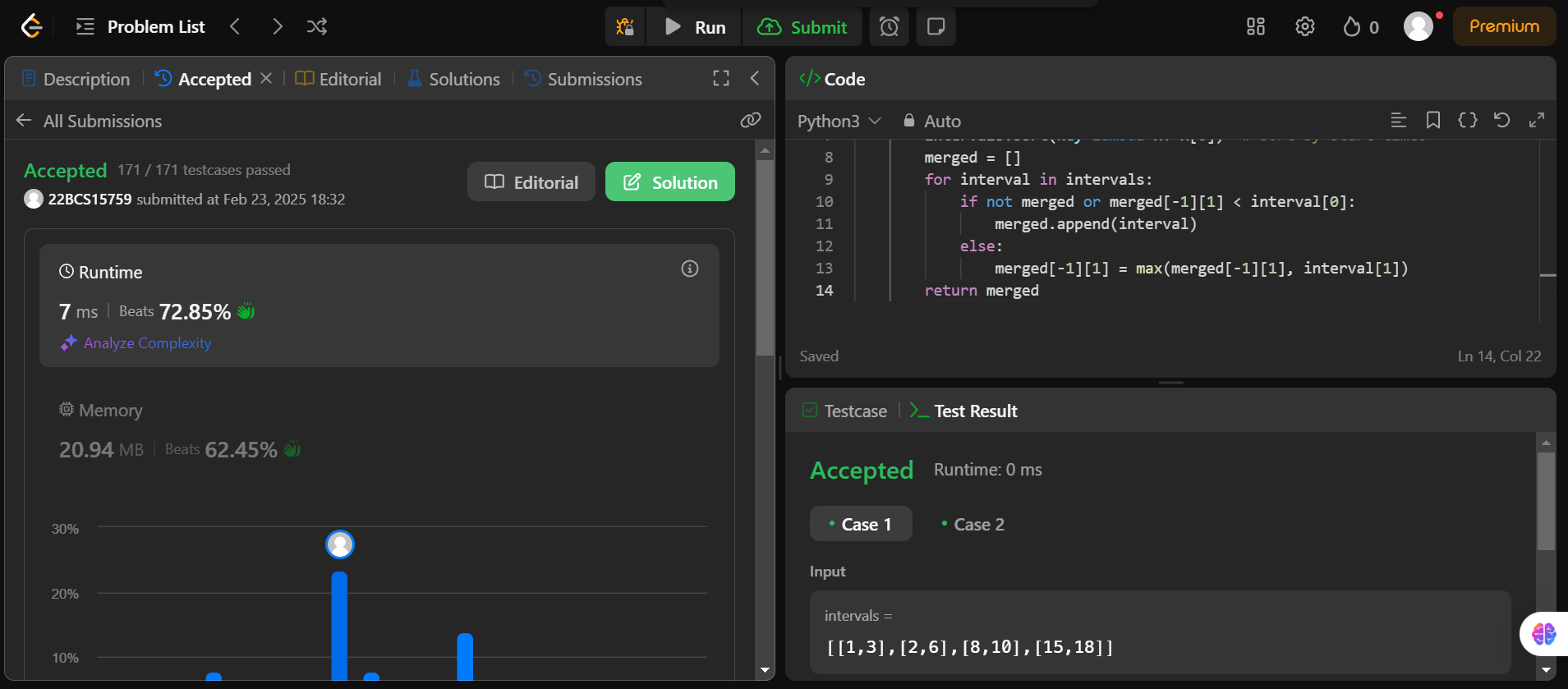
merged.append(interval)

else:

merged[-1][1] = max(merged[-1][1], interval[1])

return merged

1. **Screenshots of outputs:**

****