**Experiment 3**

**Student Name:Yash**   **UID:**22BET10301

**Branch:** BE -IT  **Section/Group:**22BET/IOT/702/A

**Semester:** 6th  **Date of Performance:**31/1/2025

**Subject Name:** Advanced Programming Lab-2  **Subject Code:** 22ITP-351

**1. Aim 1: Merge Two Sorted Lists :**

You 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.

## 2. Remove Duplicates from Sorted List II :

Given the head of a sorted linked list, delete all nodes that have duplicate numbers, leaving only distinct numbers from the original list. Return the linked list sorted as well.

**3. Merge k sorted lists:** You are given an array of k linked-lists lists, each linked-list is sorted in ascending order.Merge all the linked-lists into one sorted linked-list and return it.

**4. Sort List HW :** Given the head of a linked list, return the list after sorting it in ascending order.

**5. Merge k sorted lists :** You are given an array of k linked-lists lists, each linked-list is sorted in ascending order.Merge all the linked-lists into one sorted linked-list and return it.

**6. Rotate a list :** Given the head of a linked list, rotate the list to the right by k places.

**3. Objective:**

* Given two sorted linked lists (list1 and list2), merge them into one sorted linked list.

* The merged list should be created by splicing together the nodes from the original lists (without creating new nodes).

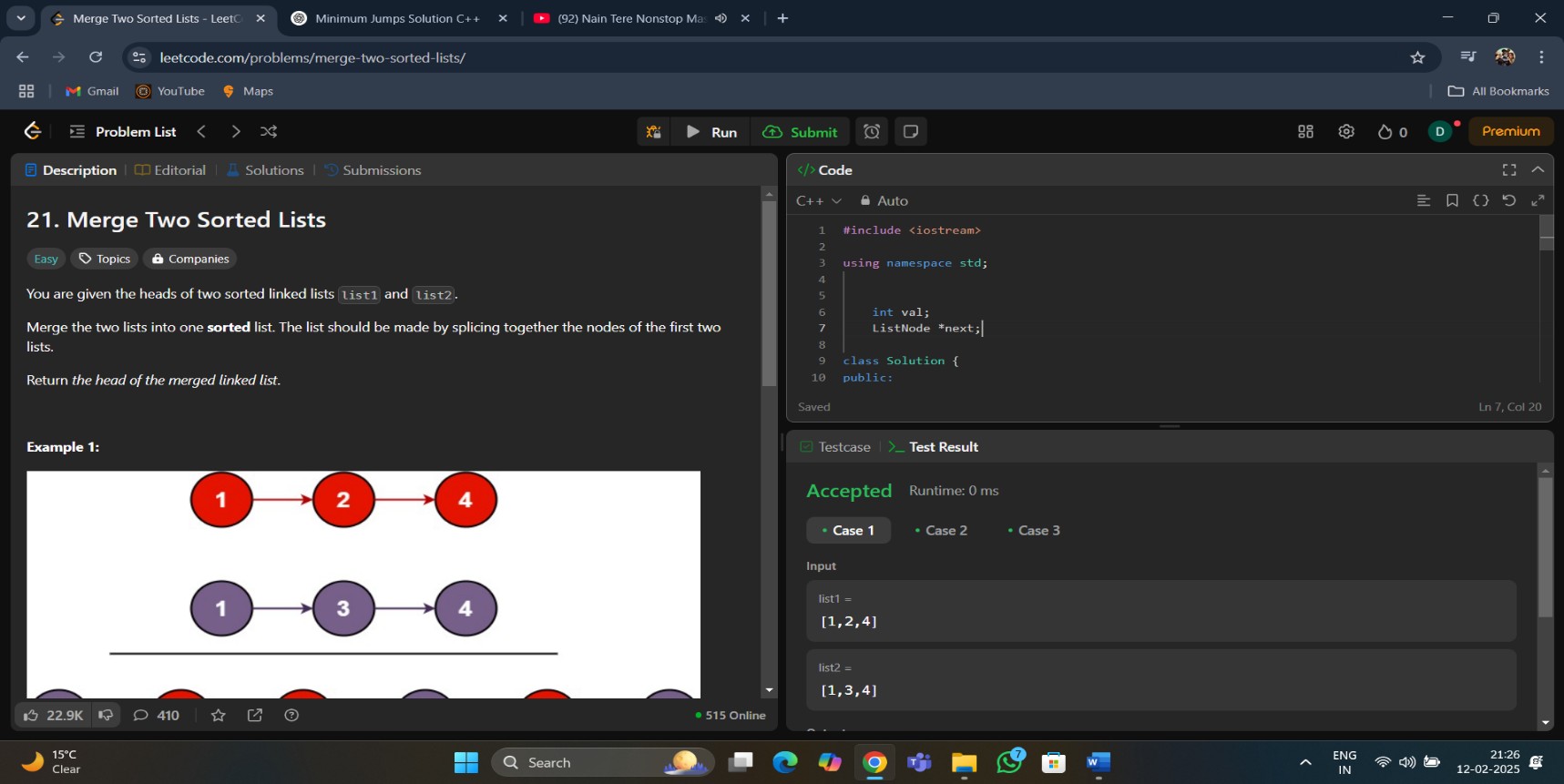
* Return the head of the new merged sorted list.
* Given a **sorted** linked list, **remove all nodes** that have duplicate values.
* Only **distinct numbers** should remain in the final linked list.
* Return the head of the modified linked list.

**4. Implementation of Code/Output 1 :**

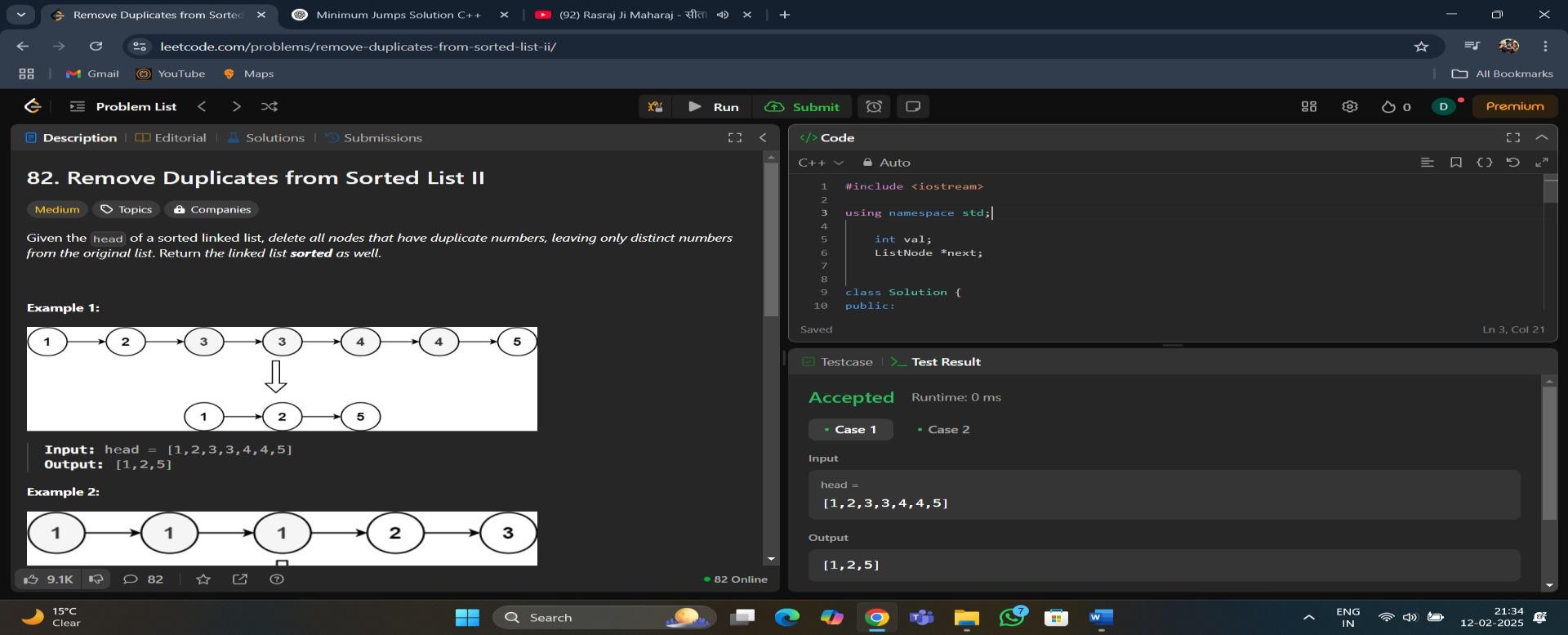
**2**

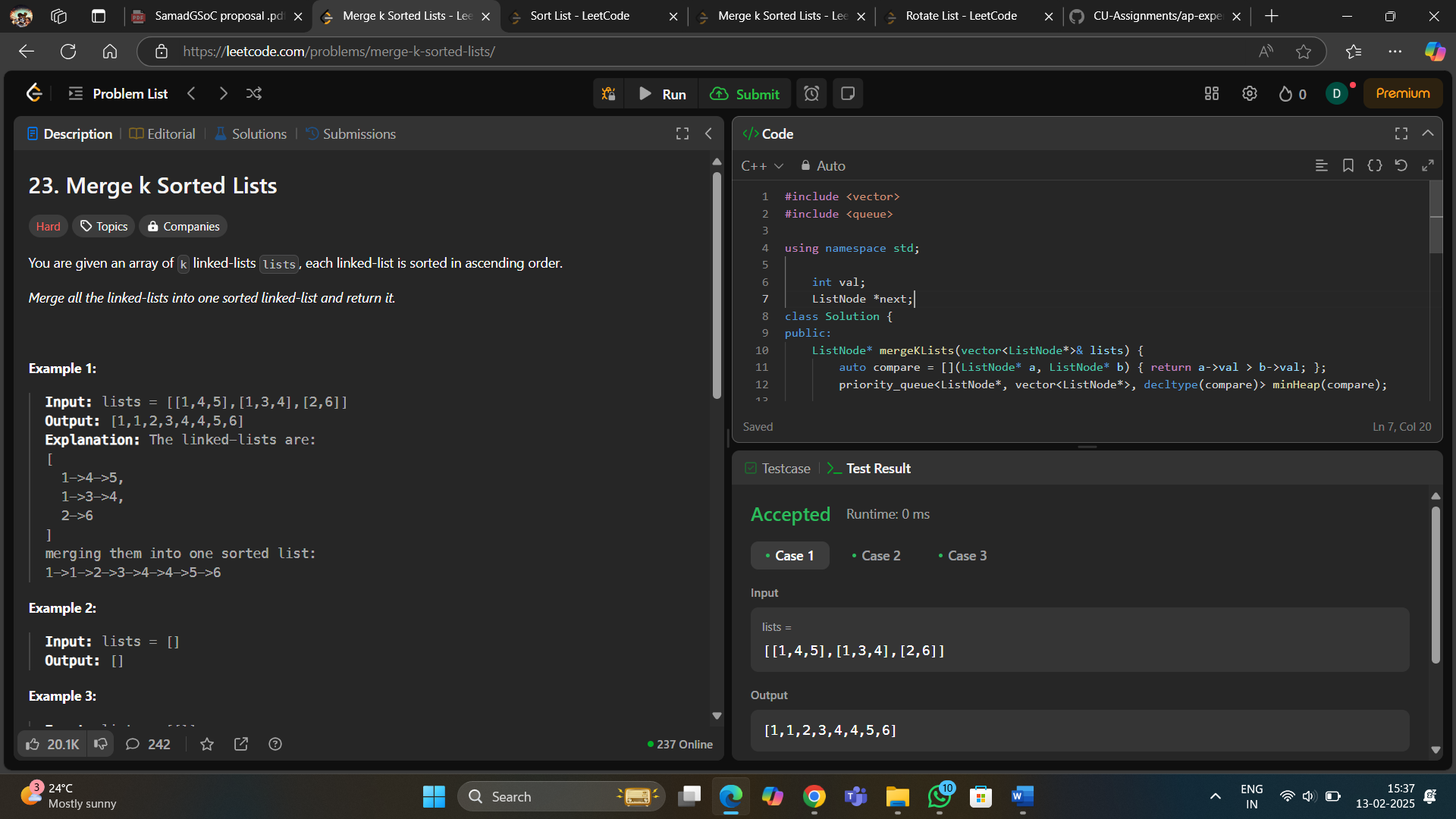
**. Code**

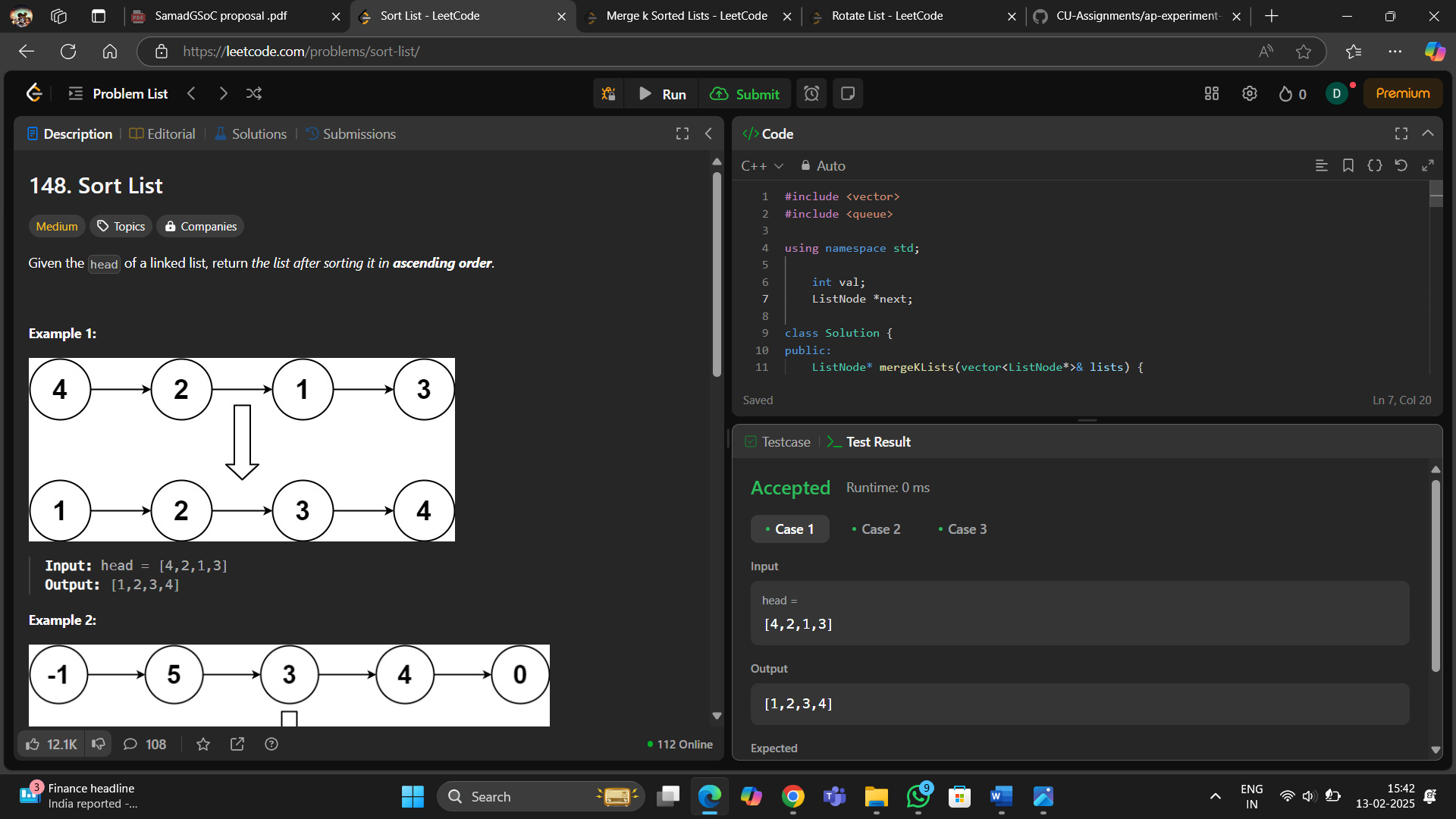
**:**

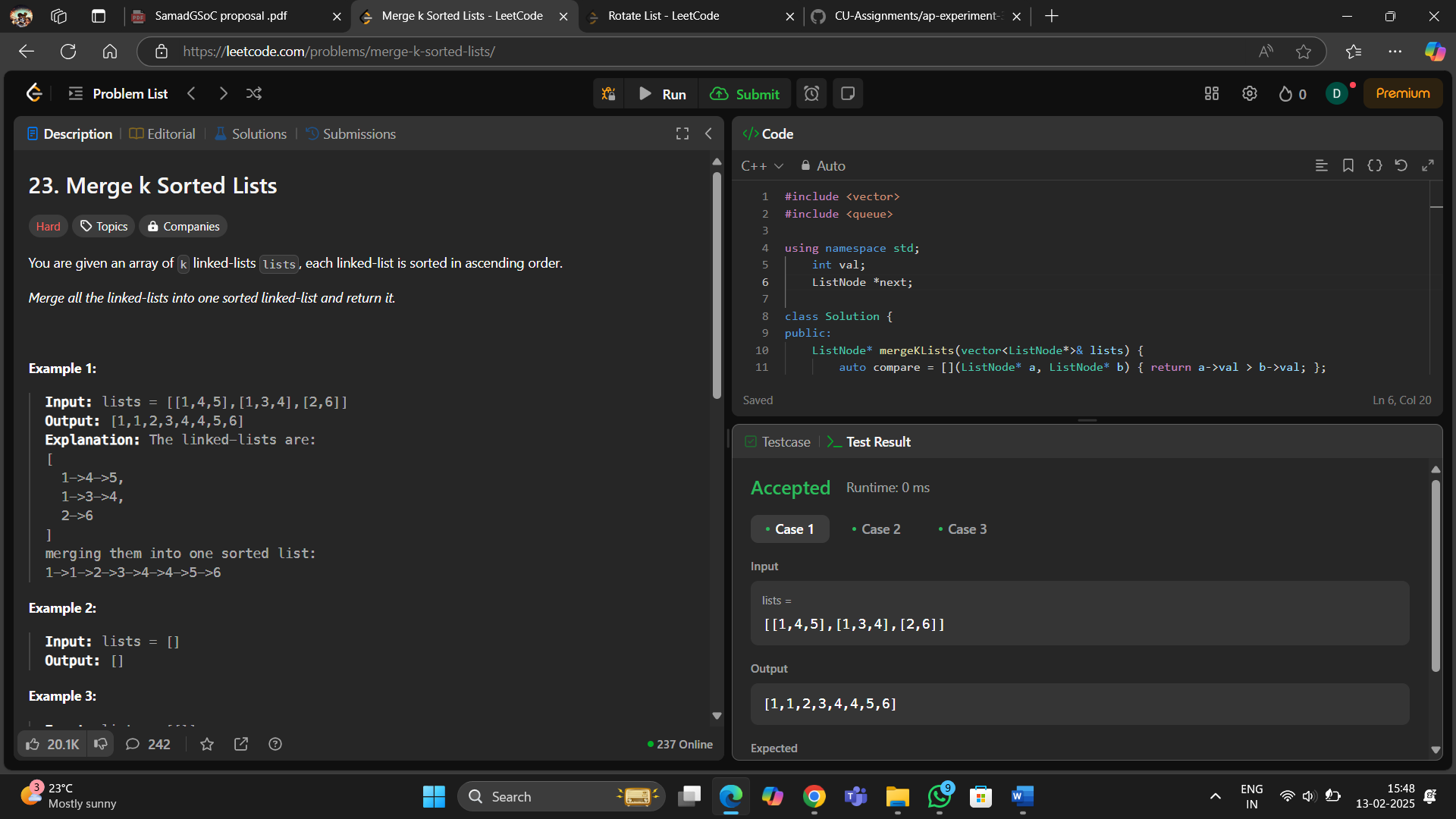


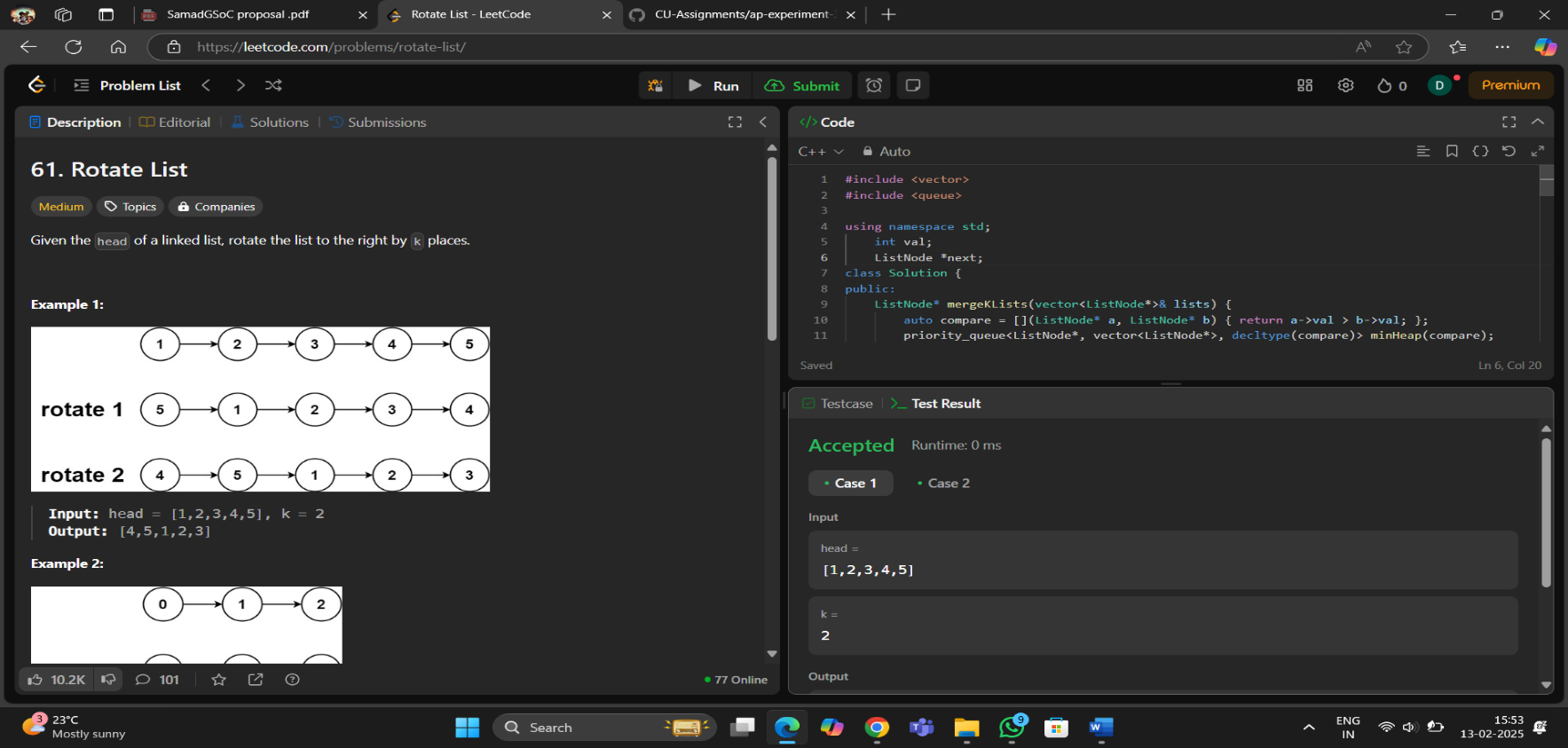
**2. Code :**



**3. code :**

**4. Code :**

**5. Code :**

**6. Code :**

**5. Learning Outcome:**

* Linked List Traversal: Iterating through multiple linked lists using pointers.
* Two-pointer technique: Used to efficiently merge two sorted lists.
* In-place Merging: Modifying the existing linked lists instead of creating new nodes.
* Dummy Node Usage: Helps simplify edge cases.
* Time Complexity Analysis: Achieves O(n) time complexity.