Date : 06/01/2023

Name : K Prasad Achari

1.Longest Substring Without Repeating Characters

Given a string s, find the length of the **longest**

**substring**

 without repeating characters.

**Example 1:**

**Input:** s = "abcabcbb"

**Output:** 3

**Explanation:** The answer is "abc", with the length of 3.

**Example 2:**

**Input:** s = "bbbbb"

**Output:** 1

**Explanation:** The answer is "b", with the length of 1.

**Example 3:**

**Input:** s = "pwwkew"

**Output:** 3

**Explanation:** The answer is "wke", with the length of 3.

Notice that the answer must be a substring, "pwke" is a subsequence and not a substring.

**Code :**

**class Solution:**

**def lengthOfLongestSubstring(self, s: str) -> int:**

**l={}**

**res=0**

**a=0**

**for i,j in enumerate(s):**

**if j not in l or l[j]<a:**

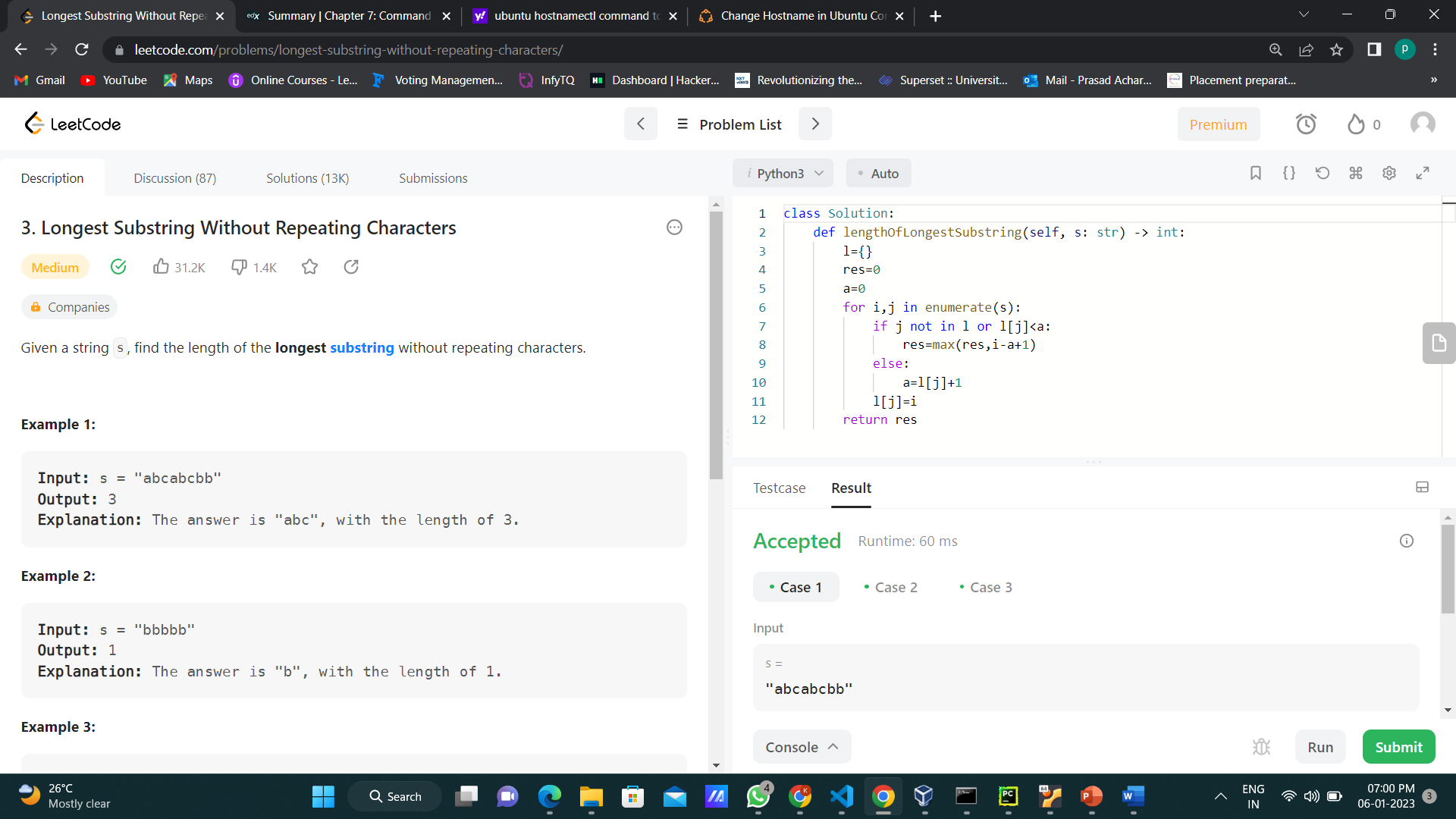
**res=max(res,i-a+1)**

**else:**

**a=l[j]+1**

**l[j]=i**

**return res**



2.Group Anagrams

Given an array of strings strs, group **the anagrams** together. You can return the answer in **any order**.

An **Anagram** is a word or phrase formed by rearranging the letters of a different word or phrase, typically using all the original letters exactly once.

**Example 1:**

**Input:** strs = ["eat","tea","tan","ate","nat","bat"]

**Output:** [["bat"],["nat","tan"],["ate","eat","tea"]]

**Example 2:**

**Input:** strs = [""]

**Output:** [[""]]

**Example 3:**

**Input:** strs = ["a"]

**Output:** [["a"]]

**Code :**

**class Solution:**

**def groupAnagrams(self, s: List[str]) -> List[List[str]]:**

**l=[]**

**d={}**

**for i in s:**

**a=''.join(sorted(list(i)))**

**if a not in l:**

**l.append(a)**

**d[a]=[i]**

**else:**

**d[a].append(i)**

**return [j for j in d.values()]**

