



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Experiment1.4

Student Name: Himanshu

Branch: BE-CSE

Semester: 6th

Subject Name: Competitive Coding-II

UID: 20BCS7944

Section/Group: 905/A

Date of Performance: 22/03/2023

Subject Code: 20CSP-351

1. Aim:

To demonstrate the concept of hashing problem.

2. Objective:

- The objective is to build problem solving capability and to learn the basic concepts of data structures.
- The implementation of missing numbers which shows and brushes up the concept of map and hashing.
- The implementation of longest substring without repeating in which the concept of hashing was introduced.

3. LeetCode code and output:

- **Longest substring without repeating characters -**

```
class Solution {
public:
    int lengthOfLongestSubstring(string s) {
        if(s.length()==0)return 0;
        unordered_map<char,int> m;

        int i=0,j=0,ans=INT_MIN;
        while(j<s.length()){
            m[s[j]]++;

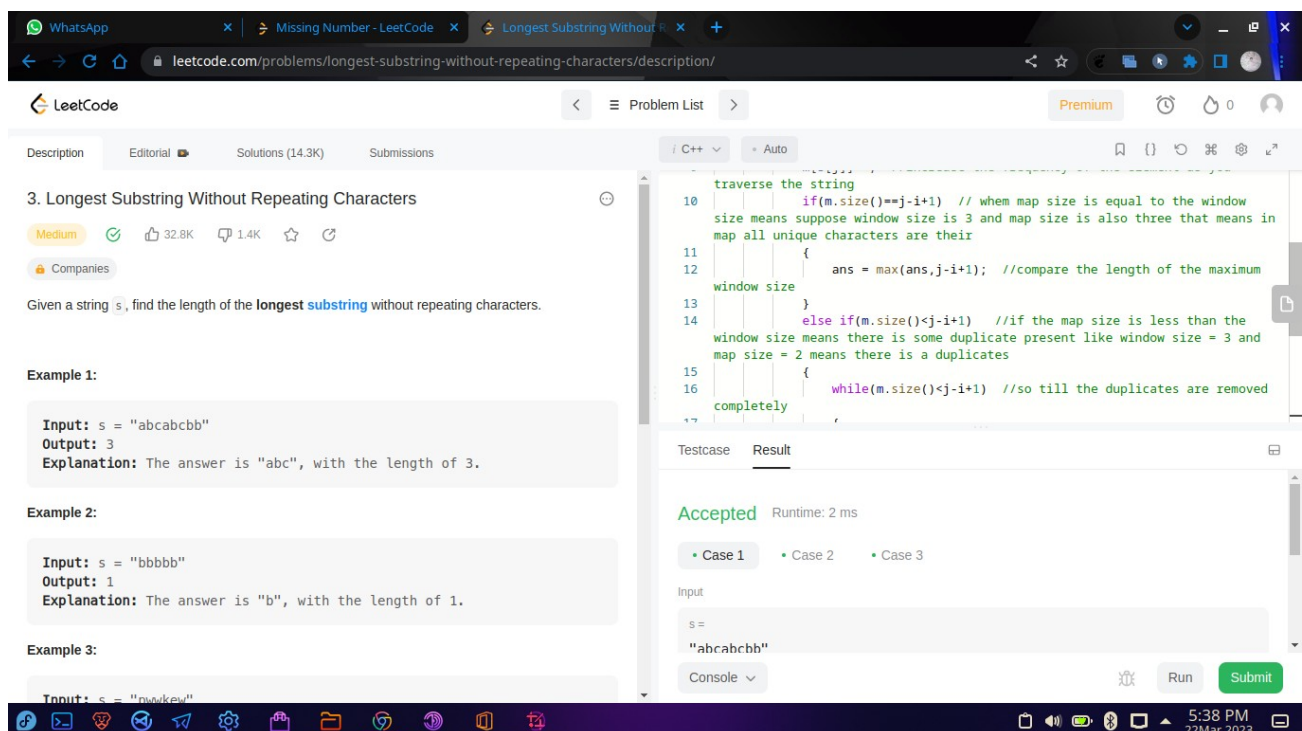
            if(m.size()==j-i+1){
                ans = max(ans,j-i+1);
            }

            else if(m.size()<j-i+1){
                while(m.size()<j-i+1){
                    m[s[i]]--;
                    if(m[s[i]]==0){
                        m.erase(s[i])
                    }
                }
            }
            j++;
        }
        return ans;
    }
};
```

```
        if(m[s[i]]==0){
            m.erase(s[i])
            i++;
        }
        j++;
    }
    return ans;

};
```

OUTPUT:



The screenshot displays a web browser window with the LeetCode website. The main content area shows the problem description for 'Longest Substring Without Repeating Characters' (Problem 3). The problem is categorized as 'Medium' and has 32.8K likes and 1.4K dislikes. The description states: 'Given a string s, find the length of the longest substring without repeating characters.' Three examples are provided: 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: 4, Explanation: The answer is "wke", with the length of 3. On the right side, the C++ solution is shown, which uses a sliding window approach with a map to track character frequencies. The code is as follows:

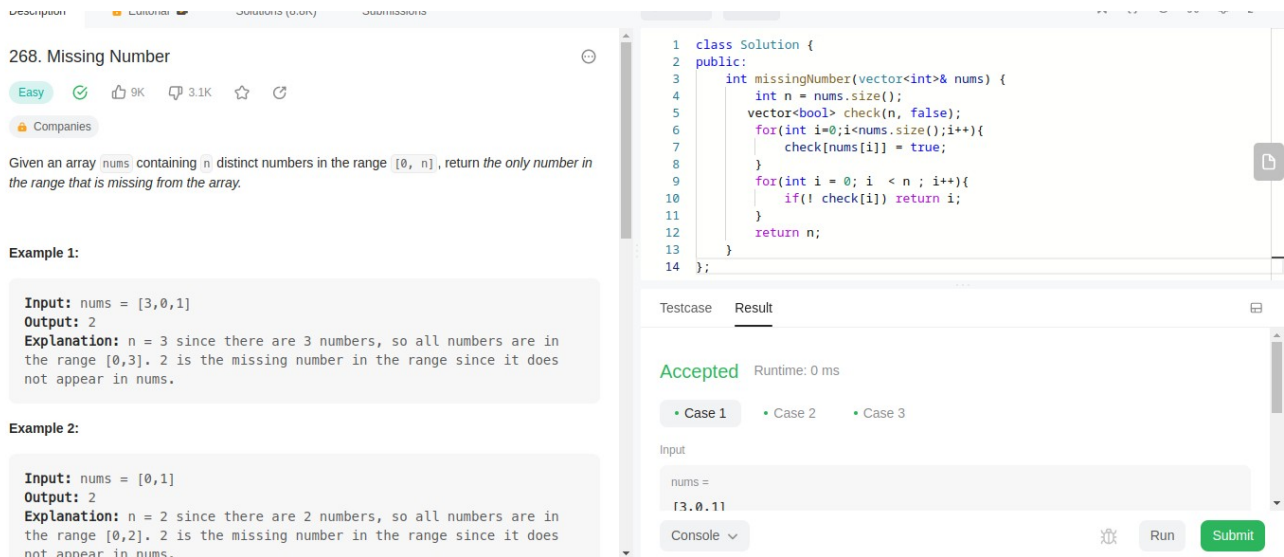
```
10 traverse the string
11     if(m.size()==j-i+1) // when map size is equal to the window
12         size means suppose window size is 3 and map size is also three that means in
13         map all unique characters are their
14     {
15         ans = max(ans,j-i+1); //compare the length of the maximum
16         window size
17     }
18     else if(m.size()<j-i+1) //if the map size is less than the
19         window size means there is some duplicate present like window size = 3 and
20         map size = 2 means there is a duplicates
21     {
22         while(m.size()<j-i+1) //so till the duplicates are removed
23             completely
24     }
```

 The 'Testcase' tab shows the result 'Accepted' with a runtime of 2 ms. The 'Input' field contains 's = "abcabcbb"'. The 'Run' and 'Submit' buttons are visible at the bottom right of the code editor.

- **Missing numbers -**

```
class Solution {
public:
    int missingNumber(vector<int>& nums) {
        int n = nums.size();
        vector<bool> check(n, false);
        for(int i=0;i<nums.size();i++){
            check[nums[i]] = true;
        }
        for(int i = 0; i < n ; i++){
            if(! check[i]) return i;
        }
        return n;
    }
};
```

OUTPUT:



The screenshot displays a coding problem titled "268. Missing Number" on a platform. The problem description states: "Given an array `nums` containing `n` distinct numbers in the range `[0, n]`, return the only number in the range that is missing from the array." It includes two examples: Example 1 with input `nums = [3,0,1]` and output `2`, and Example 2 with input `nums = [0,1]` and output `2`. The solution code is shown in a text editor, implementing a boolean check array. The code is as follows:

```
1 class Solution {
2 public:
3     int missingNumber(vector<int>& nums) {
4         int n = nums.size();
5         vector<bool> check(n, false);
6         for(int i=0;i<nums.size();i++){
7             check[nums[i]] = true;
8         }
9         for(int i = 0; i < n ; i++){
10             if(! check[i]) return i;
11         }
12         return n;
13     }
14 };
```

The solution is marked as "Accepted" with a runtime of 0 ms. The input field shows `nums = [3,0,1]` and the console output is `2`. Buttons for "Run" and "Submit" are visible at the bottom right.