DSA Mock Test – 1

Que1:- Move Zeroes

Given an integer array nums, move all 0's to the end of it while maintaining the relative order of the non-zero elements.

Note that you must do this in-place without making a copy of the array.

Example 1:
Input: nums = [0,1,0,3,12]
Output: [1,3,12,0,0]

Example 2:
Input: nums = [0]
Output: [0]

Constraints:
a. 1 <= nums.length <= 10^4
b. -2^31 <= nums[i] <= 2^31 - 1

Sol:-

```
// Time Complexity: O(n)
// Space Complexity: O(1)
class Solution {
    public void moveZeroes(int[] nums) {
        int n = nums.length;
        int right = 0, left = 0;
        while(right < n){
            if(nums[right]==0){
                right++;
            }
            else if(n > 0){
                int temp = nums[left];
                nums[left] = nums[right];
                nums[right] = temp;
                left++;
                right++;
            }
        }
    }
}
```

Que2:- First Unique Character in a String

Given a string s, find the first non-repeating character in it and return its index. If it does not

exist, return -1.

Example 1:
Input: s = "leetcode"
Output: 0

Example 2:
Input: s = "loveleetcode"
Output: 2

Example 3:
Input: s = "aabb"
Output: -1

Constraints:
a. 1 <= s.length <= 10^5
b. s consists of only lowercase English letters.

Sol:-

```
// Time Complexity: O(n)
// Space Complexity: O(n)
class Solution {
    public int firstUniqChar(String s) {
        int[] freq = new int[26];
        char[] chars = s.toCharArray();
        for(char c:chars){
            freq[c - 'a']++;
        }
        for(int i = 0; i <chars.length; i++){
            if(freq[chars[i] - 'a'] == 1){
                return i;
            }
        }
        return -1;
    }
}
```