Weekly Contest - 9

423. Reconstruct Original Digits from English

Solved

Medium | Topics | Companies

Given a string `s` containing an out-of-order English representation of digits `0-9`, return *the digits in **ascending** order*.

Example 1:

> **Input:** s = "owoztneoer"
> **Output:** "012"

Example 2:

> **Input:** s = "fviefuro"
> **Output:** "45"

Constraints:

- `1 <= s.length <= 10^5`
- `s[i]` is one of the characters `["e","g","f","i","h","o","n","s","r","u","t","w","v","x","z"]`.
- `s` is **guaranteed** to be valid.

```java
public static String originalDigits(String s) {
    int[] charFreq = new int[26];
    for (char c : s.toCharArray())
        charFreq[c-'a']++;

    int[] digCount = new int[10];
    // unique characters against the digit
    digCount[0] = charFreq['z'-'a']; // zero
    digCount[2] = charFreq['w'-'a']; // two
    digCount[4] = charFreq['u'-'a']; // four
    digCount[6] = charFreq['x'-'a']; // six
    digCount[8] = charFreq['g'-'a']; // eight

    // derived numbers
    digCount[1] = charFreq['o'-'a'] - digCount[0] - digCount[2] - digCount[4];
    digCount[3] = charFreq['h'-'a'] - digCount[8]; // three
    digCount[5] = charFreq['f'-'a'] - digCount[4]; // five
    digCount[7] = charFreq['s'-'a'] - digCount[6]; // seven
    digCount[9] = (charFreq['n'-'a'] - digCount[1] - digCount[7])/2;

    StringBuilder sb = new StringBuilder();
    for (int i=0; i<10; i++)
        while (digCount[i]-- > 0)
            sb.append(i);

    return sb.toString();
}
```

A A B A B B A K: 2 frequency
A: 2
B: 3

↳ max frequency
B = 3

Replace to have the same letter
= length of subarray - max frequency
= 5 - 3 = 2 ≤ 2

424. Longest Repeating Character Replacement

Solved

Medium | Topics | Companies

You are given a string `s` and an integer `k`. You can choose any character of the string and change it to any other uppercase English character. You can perform this operation at most `k` times.

Return *the length of the longest substring containing the same letter you can get after performing the above operations.*

Example 1:

> **Input:** s = "ABAB", k = 2
> **Output:** 4
> **Explanation:** Replace the two 'A's with two 'B's or vice versa.

Example 2:

> **Input:** s = "AABABBA", k = 1
> **Output:** 4
> **Explanation:** Replace the one 'A' in the middle with 'B' and form "AABBBBA".
> The substring "BBBB" has the longest repeating letters, which is 4.
> There may exists other ways to achieve this answer too.

Constraints:

- `1 <= s.length <= 10^5`
- `s` consists of only uppercase English letters.
- `0 <= k <= s.length`

```java
public static int characterReplacement(String s, int k) {
    int length = s.length();
    int[] freq = new int[26];
    int left = 0, right = 0, maxFreq = 0;
    int answer = 0;

    while (right < length) {
        int idx = s.charAt(right)-'A';
        freq[idx]++;
        maxFreq = Math.max(maxFreq, freq[idx]);
        // In current window, the count of characters
        // other than the max one should be ≤ k
        while (right-left+1 - maxFreq > k) {
            freq[s.charAt(left)-'A']--;
            left++;
        }
        answer = Math.max(answer, right-left+1);
        right++;
    } return answer;
}
```