Anagram

solved by	Senan
Platform	GeeksForGeeks
difficulty	Easy
tags	Hash Map
language	C++
solved on	@30/11/2024
link	https://www.geeksforgeeks.org/problems/anagram-1587115620/1
Completion	✓

Intuition

Approach

Complexity

Time Complexity:

Space Complexity:

Code

```
```

Intuition

The idea is to check if two strings are anagrams by comparing the frequency of each character in both strings. If both strings have the same character frequencies, they are anagrams; otherwise, they are not.

Approach

1. **Frequency Counting:** For each string, we maintain an array (`frequency`) of size 26 (for each letter from 'a' to 'z'). This array will store the number of times each character appears in the string.
2. **Comparison:** The strings are anagrams if their frequency arrays are identical. We compute the frequency of characters for both strings and compare them. If they match, the strings are anagrams; otherwise, they are not.

Complexity

Time Complexity:

- The time complexity is **O(n)**, where `n` is the length of the strings. This is because we iterate through each string once to calculate the frequency of characters, which takes linear time.

Space Complexity:

- The space complexity is **O(1)** because the frequency array is of constant size (26 elements, one for each lowercase letter), which does not depend on the input size.

Code

```
class Solution {
  public:
    vector<int> getFrequency(string& s){
        vector<int> frequency(26, 0);
        for(auto ch: s) frequency[ch - 'a']++;
        return frequency;
    }
    bool areAnagrams(string& s1, string& s2) {
        return getFrequency(s1) == getFrequency(s2);
    }
};
```