242. Valid Anagram

Easy 11.1K 352

Companies

Given two strings `s` and `t`, return `true` *if* `t` *is an anagram of* `s`, *and* `false` *otherwise*.

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: s = "anagram", t = "nagaram"
Output: true
```

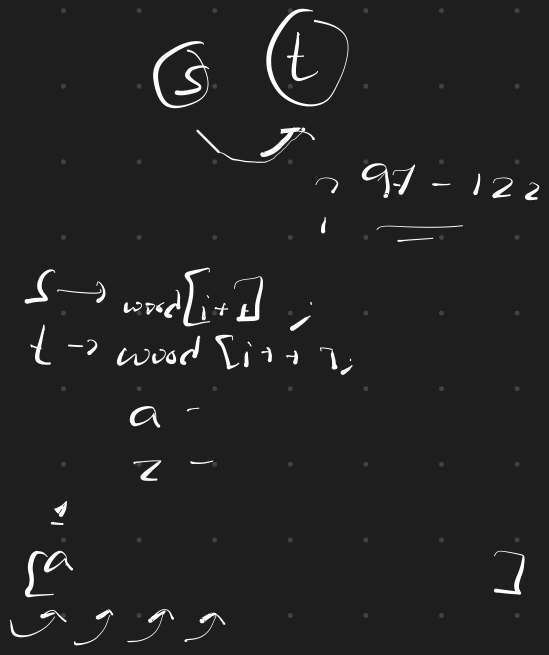
Example 2:

```
Input: s = "rat", t = "car"
Output: false
```

Constraints:

- `1 <= s.length, t.length <= 5 * 10^4`
- `s` and `t` consist of lowercase English letters.

Follow up: What if the inputs contain Unicode characters? How would you adapt your solution to such a case?



The first aproach.

(1) Take only one array to count the frequency of each element of string1.

(2) Decrement immediately, string2 mapping in that array.

(3) If the numbers 0, then OK (freq)

```
C++ | Auto
class Solution {
public:
    bool isAnagram(string s, string t) {
        int n=s.length();
        int m=t.length();
        if(n!=m)
        return false;
        int map[26] = {0};
        for(int i=0;i<n;++i)
        {
            map[s[i]-'a']++;
            map[t[i]-'a']--;
        }
        for(int i:map){
            if(i!=0)
            return false;
        }
        return true;
    }
};
```

The second aproach

(1) Sort both the strings.

(2) Check whether two strings are equal.

```
class Solution {
public:
    bool isAnagram(std::string s, std::string t) {
        // Convert strings to char arrays
        std::sort(s.begin(), s.end());
        std::sort(t.begin(), t.end());

        // Check if the sorted char arrays are equal
        return s == t;
    }
};
```

Previous Next