💡LeetCode 242 - *Valid Anagram*

Problem link : 🔗 Problem Link

🚀 Objective:

Determine if two strings are anagrams of each other.

An **anagram** is formed by rearranging all characters of one string to form another, using each character exactly once.

🧠 Approach: Sorting-based Comparison

```
class Solution {
public:
    bool isAnagram(string s, string t) {
        sort(s.begin(), s.end());   // Sort string s alphabetically
        sort(t.begin(), t.end());   // Sort string t alphabetically
        return s == t;              // Compare both sorted strings
    }
};
```

📝 Explanation:

1. **Sorting Step**:
    - Sort both input strings `s` and `t`.
    - If both strings are anagrams, their sorted versions will be exactly the same.
2. **Comparison Step**:
    - If sorted `s` is equal to sorted `t`, return `true`.

- Otherwise, return `false`.

⏱️ Time & Space Complexity:

Complexity	Value
Time	O(n log n)
Space (Auxiliary)	O(1) *(ignoring sort space)*

> Where n is the length of the input strings.

✅ Sample Input/Output:

Input	Output
`s = "anagram", t = "nagaram"`	`true`
`s = "rat", t = "car"`	`false`

📎 Notes:

- Works perfectly for lowercase alphabets.
- Can be optimized further using frequency count (O(n) approach).
- This version is clean, readable, and great for interviews or learning!
