

## 451. Sort Characters By Frequency

Solved

Medium Topics Companies

Given a string `s`, sort it in **decreasing order** based on the **frequency** of the characters. The **frequency** of a character is the number of times it appears in the string.

Return *the sorted string*. If there are multiple answers, return *any of them*.

### Example 1:

**Input:** `s = "tree"`

**Output:** `"eert"`

**Explanation:** 'e' appears twice while 'r' and 't' both appear once. So 'e' must appear before both 'r' and 't'. Therefore "eetr" is also a valid answer.

### Example 2:

**Input:** `s = "cccaaa"`

**Output:** `"aaaccc"`

**Explanation:** Both 'c' and 'a' appear three times, so both "cccaaa" and "aaaccc" are valid answers.

Note that "cacaca" is incorrect, as the same characters must be together.

### Example 3:

**Input:** `s = "Aabb"`

**Output:** `"bbaa"`

The screenshot displays the LeetCode problem page for '451. Sort Characters By Frequency'. The left sidebar shows the problem description and a submission status of 'Accepted' for user 'ZhukovIlya' at Jul 07, 2024 16:35. The submission details show a runtime of 58 ms (Beats 88.20%) and memory usage of 44.55 MB (Beats 30.29%). A bar chart shows the runtime distribution. The main area displays the C# code for the solution, which uses a Dictionary to count character frequencies and sorts the string based on these frequencies. The right sidebar shows the 'Testcase' and 'Test Result' for Case 1, where the input is 'tree' and the output is 'eetr', with a runtime of 66 ms.

```

using System;
using System.Collections.Generic;
using System.Linq;

public class Solution {
    public string FrequencySort(string s) {
        // Словарь для подсчета частоты символов
        Dictionary<char, int> frequencyMap = new Dictionary<char, int>();
        foreach (char c in s) {
            if (frequencyMap.ContainsKey(c)) {
                frequencyMap[c]++;
            }
        }
        // ... (rest of the code)
    }
}
    
```

Код:

```

using System;
using System.Collections.Generic;
using System.Linq;

public class Solution {
    
```

```

{
    public string FrequencySort(string s)
    {
        // Словарь для подсчета частоты символов
        Dictionary<char, int> frequencyMap = new Dictionary<char, int>();
        foreach (char c in s)
        {
            if (frequencyMap.ContainsKey(c))
            {
                frequencyMap[c]++;
            }
            else
            {
                frequencyMap[c] = 1;
            }
        }

        // Сортировка символов по частоте
        var sortedChars = frequencyMap.OrderByDescending(pair => pair.Value)
            .SelectMany(pair => new string(pair.Key,
pair.Value));

        // Объединение отсортированных символов в строку
        return new string(sortedChars.ToArray());
    }
}

```