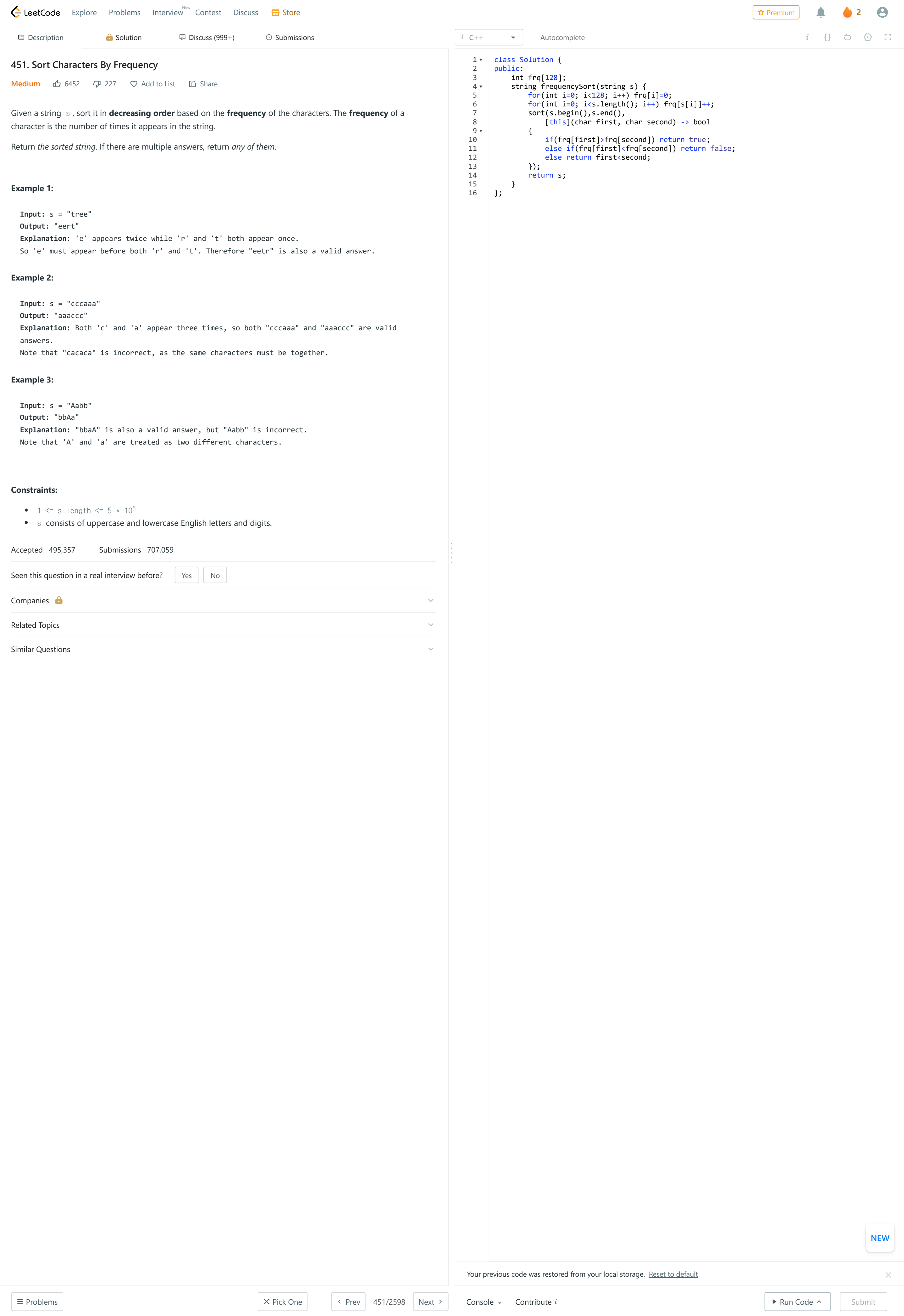

451. Sort Characters By Frequency

Medium 6452 227 Add to List Share

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"
Explanation: "bbaA" is also a valid answer, but "Aabb" is incorrect.
Note that 'A' and 'a' are treated as two different characters.
```

Constraints:

- `1 <= s.length <= 5 * 10`^{5}
- `s` consists of uppercase and lowercase English letters and digits.

Accepted 495,357 Submissions 707,059

Seen this question in a real interview before? Yes No

Companies

Related Topics

Similar Questions

```cpp
class Solution {
public:
    int frq[128];
    string frequencySort(string s) {
        for(int i=0; i<128; i++) frq[i]=0;
        for(int i=0; i<s.length(); i++) frq[s[i]]++;
        sort(s.begin(),s.end(),
            [this](char first, char second) -> bool
        {
            if(frq[first]>frq[second]) return true;
            else if(frq[first]<frq[second]) return false;
            else return first<second;
        });
        return s;
    }
};
```

Your previous code was restored from your local storage. Reset to default