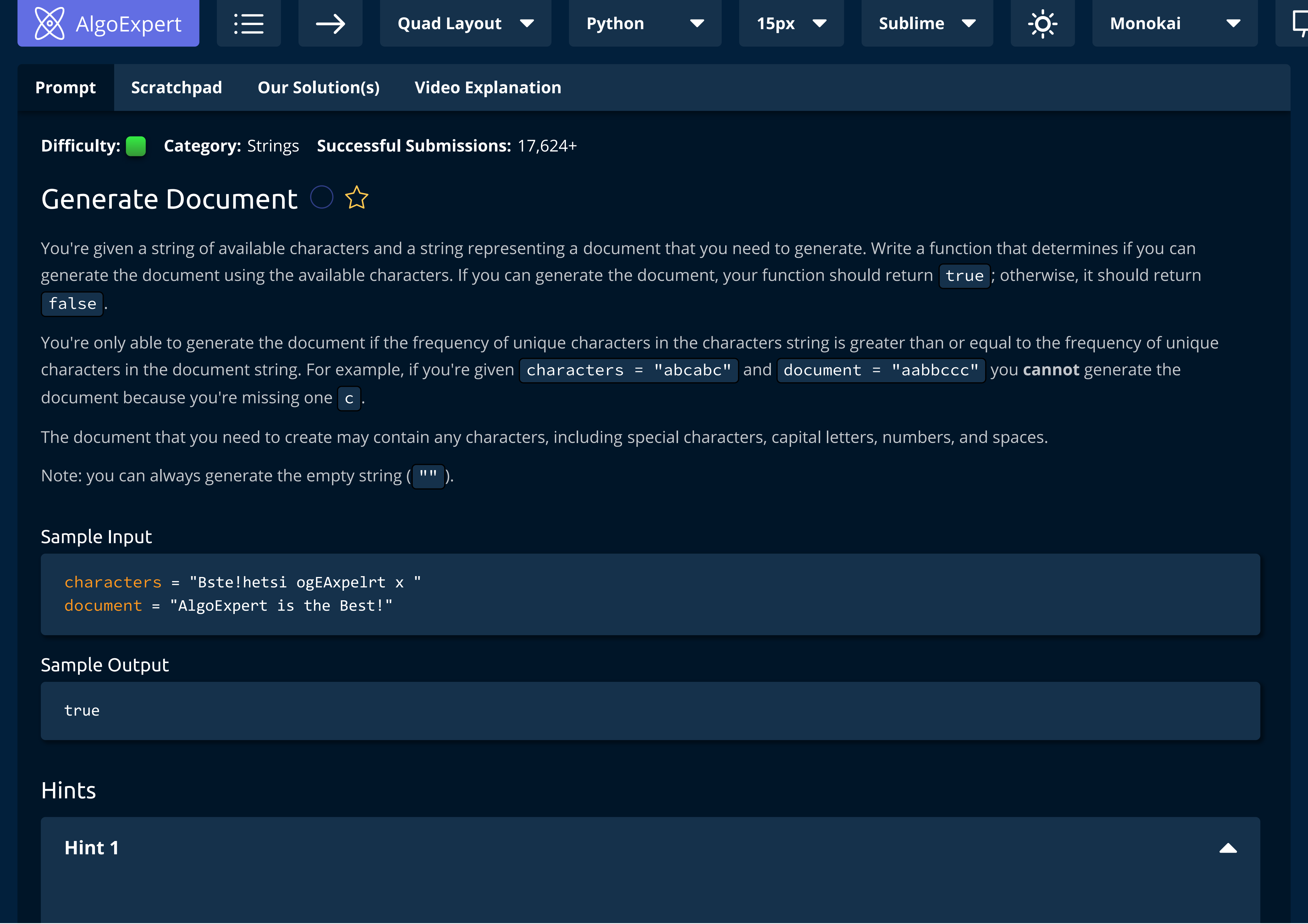


Difficulty: 🟩 **Category:** Strings **Successful Submissions:** 17,624+

Generate Document ○ ☆

You're given a string of available characters and a string representing a document that you need to generate. Write a function that determines if you can generate the document using the available characters. If you can generate the document, your function should return `true`; otherwise, it should return `false`.

You're only able to generate the document if the frequency of unique characters in the characters string is greater than or equal to the frequency of unique characters in the document string. For example, if you're given `characters = "abcabc"` and `document = "aabbccc"` you **cannot** generate the document because you're missing one `c`.

The document that you need to create may contain any characters, including special characters, capital letters, numbers, and spaces.

Note: you can always generate the empty string (`""`).

Sample Input

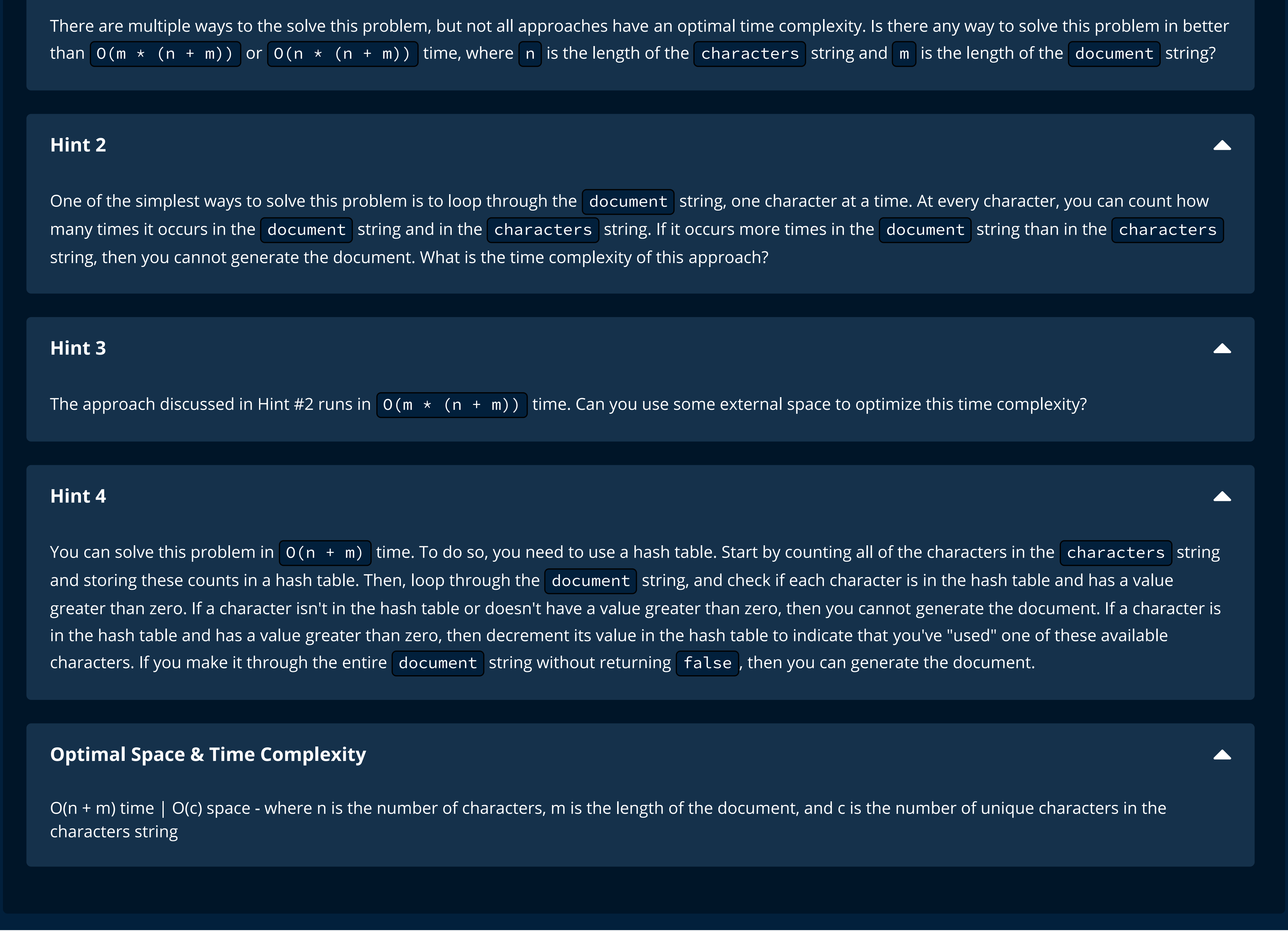
```
characters = "Bste!hetsi ogEAxpelrt x "
document = "AlgoExpert is the Best!"
```

Sample Output

```
true
```

Hints

Hint 1



There are multiple ways to the solve this problem, but not all approaches have an optimal time complexity. Is there any way to solve this problem in better than `O(m * (n + m))` or `O(n * (n + m))` time, where `n` is the length of the `characters` string and `m` is the length of the `document` string?

Hint 2

One of the simplest ways to solve this problem is to loop through the `document` string, one character at a time. At every character, you can count how many times it occurs in the `document` string and in the `characters` string. If it occurs more times in the `document` string than in the `characters` string, then you cannot generate the document. What is the time complexity of this approach?

Hint 3

The approach discussed in Hint #2 runs in `O(m * (n + m))` time. Can you use some external space to optimize this time complexity?

Hint 4

You can solve this problem in `O(n + m)` time. To do so, you need to use a hash table. Start by counting all of the characters in the `characters` string and storing these counts in a hash table. Then, loop through the `document` string, and check if each character is in the hash table and has a value greater than zero. If a character isn't in the hash table or doesn't have a value greater than zero, then you cannot generate the document. If a character is in the hash table and has a value greater than zero, then decrement its value in the hash table to indicate that you've "used" one of these available characters. If you make it through the entire `document` string without returning `false`, then you can generate the document.

Optimal Space & Time Complexity

O(n + m) time | O(c) space - where n is the number of characters, m is the length of the document, and c is the number of unique characters in the characters string