Find one extra character in string

There are a few different approaches you could take to find the extra character in a string in C++, depending on the specifics of the problem. Here is one approach you could take:

1. Create two arrays, one to store the count of characters in the first string, and the other to store the count of characters in the second string.

2. Iterate through each character in the first string, and use the character as an index in the first array to increment the count at that index.

3. Iterate through each character in the second string, and use the character as an index in the second array to increment the count at that index.

4. Iterate through the two arrays, comparing the count at each index. If the count is different between the two arrays at a given index, then the character at that index is the extra character.

Here is an example of what the code for this approach might look like:

```cpp
#include <iostream>
#include <string>

using namespace std;

string findExtraCharacter(string s1, string s2) {
    int count1[256] = {0}, count2[256] = {0}; // create two a

    // iterate through each character in the first string, in
    for (int i = 0; i < s1.length(); i++) {
        count1[s1[i]]++;
    }

    // iterate through each character in the second string, i
    for (int i = 0; i < s2.length(); i++) {
        count2[s2[i]]++;
    }

    // iterate through both arrays, comparing the count at ea
    for (int i = 0; i < 256; i++) {
```

```cpp
        if (count1[i] != count2[i]) {
            // if the count is different between the two arra
            return std::string(1, i);
        }
    }
    return "";
}

int main() {
    string s1 = "hello";
    string s2 = "hefloo";
    cout << "The extra character in s2 is: " << findExtraChar
    return 0;
}
```

OUTPUT:

```
The extra character in s2 is: l
```

```
Please note that this approach only finds one extra charact
er present in the string,if there are more than one charact
ers which are extra in the second string then it would not
work.
Another point to notice is that this is assuming the input
strings are english alphabets based and for other kind of c
haracters this approach may not work.
```