

Problem 3104: Find Longest Self-Contained Substring

Problem Information

Difficulty: Hard

Acceptance Rate: 58.71%

Paid Only: Yes

Tags: Hash Table, String, Binary Search, Prefix Sum

Problem Description

Given a string `s`, your task is to find the length of the **longest self-contained** substring of `s`.

A substring `t` of a string `s` is called **self-contained** if `t != s` and for every character in `t`, it doesn't exist in the rest of `s`.

Return the length of the longest self-contained substring of `s` if it exists, otherwise, return -1.

Example 1:

Input: s = "abba"

Output: 2

Explanation: Let's check the substring `bb`. You can see that no other "b" is outside of this substring. Hence the answer is 2.

Example 2:

Input: s = "abab"

Output: -1

Explanation: Every substring we choose does not satisfy the described property (there is some character which is inside and outside of that substring). So the answer would be -1.

Example 3:

Input: s = "abacd"

Output: 4

Explanation: Let's check the substring `"abac"`. There is only one character outside of this substring and that is `"d"`. There is no `"d"` inside the chosen substring, so it satisfies the condition and the answer is 4.

Constraints:

* `2 <= s.length <= 5 * 104` * `s` consists only of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
    int maxSubstringLength(string s) {
        }
    };
}
```

Java:

```
class Solution {
public int maxSubstringLength(String s) {
        }
    }
}
```

Python3:

```
class Solution:
    def maxSubstringLength(self, s: str) -> int:
```

