Problem 3713: Longest Balanced Substring I

Problem Information

Difficulty: Medium
Acceptance Rate: 52.36%
Paid Only: No
Tags: Hash Table, String, Counting, Enumeration

Problem Description

You are given a string `s` consisting of lowercase English letters.

A **substring** of `s` is called **balanced** if all **distinct** characters in the **substring** appear the **same** number of times.

Return the **length** of the **longest balanced substring** of `s`.

Example 1:

Input: s = "abbac"

Output: 4

Explanation:

The longest balanced substring is `"abba"` because both distinct characters `'a'` and `'b'` each appear exactly 2 times.

Example 2:

Input: s = "zzabccy"

Output: 4

Explanation:

The longest balanced substring is `"zabc"` because the distinct characters `'z'`, `'a'`, `'b'`, and `'c'` each appear exactly 1 time.

Example 3:

Input: s = "aba"

Output: 2

Explanation:

■■■■■■■ One of the longest balanced substrings is `"ab"` because both distinct characters `'a'` and `'b'` each appear exactly 1 time. Another longest balanced substring is `"ba"`.

Constraints:

* `1 <= s.length <= 1000` * `s` consists of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
int longestBalanced(string s) {

}
};
```

Java:

```
class Solution {
public int longestBalanced(String s) {

}
}
```

Python3:

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