

Problem 2609: Find the Longest Balanced Substring of a Binary String

Problem Information

Difficulty: Easy

Acceptance Rate: 46.00%

Paid Only: No

Tags: String

Problem Description

You are given a binary string `s` consisting only of zeroes and ones.

A substring of `s` is considered balanced if**all zeroes are before ones** and the number of zeroes is equal to the number of ones inside the substring. Notice that the empty substring is considered a balanced substring.

Return _the length of the longest balanced substring of_ `s` .

A **substring** is a contiguous sequence of characters within a string.

Example 1:

Input: s = "01000111" **Output:** 6 **Explanation:** The longest balanced substring is "000111", which has length 6.

Example 2:

Input: s = "00111" **Output:** 4 **Explanation:** The longest balanced substring is "0011", which has length 4.

Example 3:

Input: s = "111" **Output:** 0 **Explanation:** There is no balanced substring except the empty substring, so the answer is 0.

****Constraints:****

* `1 <= s.length <= 50` * `0 <= s[i] <= 1`

Code Snippets

C++:

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

Java:

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

Python3:

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