Problem 3499: Maximize Active Section with Trade I

Problem Information

Difficulty: Medium
Acceptance Rate: 30.82%
Paid Only: No
Tags: String, Enumeration

Problem Description

You are given a binary string `s` of length `n`, where:

* `'1'` represents an **active** section. * `'0'` represents an **inactive** section.

You can perform **at most one trade** to maximize the number of active sections in `s`. In a trade, you:

* Convert a contiguous block of `'1'`s that is surrounded by `'0'`s to all `'0'`s. * Afterward, convert a contiguous block of `'0'`s that is surrounded by `'1'`s to all `'1'`s.

Return the **maximum** number of active sections in `s` after making the optimal trade.

Note: Treat `s` as if it is **augmented** with a `'1'` at both ends, forming `t = '1' + s + '1'`. The augmented `'1'`s **do not** contribute to the final count.

Example 1:

Input: s = "01"

Output: 1

Explanation:

Because there is no block of `'1'`s surrounded by `'0'`s, no valid trade is possible. The maximum number of active sections is 1.

Example 2:

Input: s = "0100"

Output: 4

Explanation:

* String `"0100"` -> Augmented to `"101001"`. * Choose `"0100"`, convert `"10 _**1**_ 001"` -> `"1 _**0000**_ 1"` -> `"1 _**1111**_ 1"`. * The final string without augmentation is `"1111"`. The maximum number of active sections is 4.

Example 3:

Input: s = "1000100"

Output: 7

Explanation:

* String `"1000100"` -> Augmented to `"110001001"`. * Choose `"000100"`, convert `"11000 _**1**_ 001"` -> `"11 _**000000**_ 1"` -> `"11 _**111111**_ 1"`. * The final string without augmentation is `"1111111"`. The maximum number of active sections is 7.

Example 4:

Input: s = "01010"

Output: 4

Explanation:

* String `"01010"` -> Augmented to `"1010101"`. * Choose `"010"`, convert `"10 _**1**_ 0101"` -> `"1 _**000**_ 101"` -> `"1 _**111**_ 101"`. * The final string without augmentation is `"11110"`. The maximum number of active sections is 4.

Constraints:

* `1 <= n == s.length <= 105` * `s[i]` is either `'0'` or `'1'`

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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