Problem 3303: Find the Occurrence of First Almost Equal Substring

Problem Information

Difficulty: Hard
Acceptance Rate: 14.75%
Paid Only: No
Tags: String, String Matching

Problem Description

You are given two strings `s` and `pattern`.

A string `x` is called **almost equal** to `y` if you can change **at most** one character in `x` to make it _identical_ to `y`.

Return the **smallest** _starting index_ of a substring in `s` that is **almost equal** to `pattern`. If no such index exists, return `-1`.

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

Example 1:

Input: s = "abcdefg", pattern = "bcdffg"

Output: 1

Explanation:

The substring `s[1..6] == "bcdefg"` can be converted to `"bcdffg"` by changing `s[4]` to `"f"`.

Example 2:

Input: s = "ababbababa", pattern = "bacaba"

Output: 4

Explanation:

The substring `s[4..9] == "bababa"` can be converted to `"bacaba"` by changing `s[6]` to `"c"`.

Example 3:

Input: s = "abcd", pattern = "dba"

Output: -1

Example 4:

Input: s = "dde", pattern = "d"

Output: 0

Constraints:

* `1 <= pattern.length < s.length <= 105` * `s` and `pattern` consist only of lowercase English letters.

Follow-up: Could you solve the problem if **at most** `k` **consecutive** characters can be changed?

Code Snippets

C++:

```
class Solution {
public:
int minStartingIndex(string s, string pattern) {

}
};
```

Java:

```
class Solution {
public int minStartingIndex(String s, String pattern) {

}
}
```

Python3:

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