Problem 1668: Maximum Repeating Substring

Problem Information

Difficulty: Easy
Acceptance Rate: 40.39%
Paid Only: No
Tags: String, Dynamic Programming, String Matching

Problem Description

For a string `sequence`, a string `word` is **`k`-repeating** if `word` concatenated `k` times is a substring of `sequence`. The `word`'s **maximum`k`-repeating value** is the highest value `k` where `word` is `k`-repeating in `sequence`. If `word` is not a substring of `sequence`, `word`'s maximum `k`-repeating value is `0`.

Given strings `sequence` and `word`, return _the**maximum`k`-repeating value** of `word` in `sequence`_.

Example 1:

Input: sequence = "ababc", word = "ab" **Output:** 2 **Explanation:** "abab" is a substring in "_abab_ c".

Example 2:

Input: sequence = "ababc", word = "ba" **Output:** 1 **Explanation:** "ba" is a substring in "a _ba_ bc". "baba" is not a substring in "ababc".

Example 3:

Input: sequence = "ababc", word = "ac" **Output:** 0 **Explanation:** "ac" is not a substring in "ababc".

Constraints:

* `1 <= sequence.length <= 100` * `1 <= word.length <= 100` * `sequence` and `word` contains only lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
int maxRepeating(string sequence, string word) {

}
};
```

Java:

```
class Solution {
public int maxRepeating(String sequence, String word) {

}
}
```

Python3:

```
class Solution:
def maxRepeating(self, sequence: str, word: str) -> int:
```