Problem 1147: Longest Chunked Palindrome Decomposition

Problem Information

Difficulty: Hard
Acceptance Rate: 59.05%
Paid Only: No
Tags: Two Pointers, String, Dynamic Programming, Greedy, Rolling Hash, Hash Function

Problem Description

You are given a string `text`. You should split it to k substrings `(subtext1, subtext2, ..., subtextk)` such that:

* `subtexti` is a **non-empty** string. * The concatenation of all the substrings is equal to `text` (i.e., `subtext1 + subtext2 + ... + subtextk == text`). * `subtexti == subtextk - i + 1` for all valid values of `i` (i.e., `1 <= i <= k`).

Return the largest possible value of `k`.

Example 1:

Input: text = "ghiabcdefhelloadamhelloabcdefghi" **Output:** 7 **Explanation:** We can split the string on "(ghi)(abcdef)(hello)(adam)(hello)(abcdef)(ghi)".

Example 2:

Input: text = "merchant" **Output:** 1 **Explanation:** We can split the string on "(merchant)".

Example 3:

Input: text = "antaprezatepzapreanta" **Output:** 11 **Explanation:** We can split the string on "(a)(nt)(a)(pre)(za)(tep)(za)(pre)(a)(nt)(a)".

Constraints:

* `1 <= text.length <= 1000` * `text` consists only of lowercase English characters.

Code Snippets

C++:

```
class Solution {
public:
int longestDecomposition(string text) {

}
};
```

Java:

```
class Solution {
public int longestDecomposition(String text) {

}
}
```

Python3:

```
class Solution:
def longestDecomposition(self, text: str) -> int:
```