Problem 3163: String Compression III

Problem Information

Difficulty: Medium
Acceptance Rate: 66.94%
Paid Only: No
Tags: String

Problem Description

Given a string `word`, compress it using the following algorithm:

* Begin with an empty string `comp`. While `word` is **not** empty, use the following operation: * Remove a maximum length prefix of `word` made of a _single character_ `c` repeating **at most** 9 times. * Append the length of the prefix followed by `c` to `comp`.

Return the string `comp`.

Example 1:

Input: word = "abcde"

Output: "1a1b1c1d1e"

Explanation:

Initially, `comp = ""`. Apply the operation 5 times, choosing `"a"`, `"b"`, `"c"`, `"d"`, and `"e"` as the prefix in each operation.

For each prefix, append `"1"` followed by the character to `comp`.

Example 2:

Input: word = "aaaaaaaaaaaaaabb"

Output: "9a5a2b"

Explanation:

Initially, `comp = ""`. Apply the operation 3 times, choosing `"aaaaaaaaa"`, `"aaaaa"`, and `"bb"` as the prefix in each operation.

* For prefix `"aaaaaaaaa"`, append `"9"` followed by `"a"` to `comp`. * For prefix `"aaaaa"`, append `"5"` followed by `"a"` to `comp`. * For prefix `"bb"`, append `"2"` followed by `"b"` to `comp`.

Constraints:

* `1 <= word.length <= 2 * 105` * `word` consists only of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
string compressedString(string word) {

}
};
```

Java:

```
class Solution {
public String compressedString(String word) {

}
}
```

Python3:

```
class Solution:
def compressedString(self, word: str) -> str:
```