6195. Maximum Deletions on a String

My Submissions (/contest/weekly-contest-313/problems/maximum-deletions-on-a-string/submissions/) | Back to Contest (/contest/weekly-contest-313/)

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Hard

You are given a string `s` consisting of only lowercase English letters. In one operation, you can:

- Delete **the entire string** `s`, or
- Delete the **first** `i` letters of `s` if the first `i` letters of `s` are **equal** to the following `i` letters in `s`, for any `i` in the range `1 <= i <= s.length / 2`.

For example, if `s = "ababc"`, then in one operation, you could delete the first two letters of `s` to get `"abc"`, since the first two letters of `s` and the following two letters of `s` are both equal to `"ab"`.

Return *the* ***maximum*** *number of operations needed to delete all of* `s`.

Example 1:

```
Input: s = "abcabcdabc"
Output: 2
Explanation:
- Delete the first 3 letters ("abc") since the next 3 letters are equal. Now, s = "abcdabc".
- Delete all the letters.
We used 2 operations so return 2. It can be proven that 2 is the maximum number of operations needed.
Note that in the second operation we cannot delete "abc" again because the next occurrence of "abc" does not happen in the next
```

Example 2:

```
Input: s = "aaabaab"
Output: 4
Explanation:
- Delete the first letter ("a") since the next letter is equal. Now, s = "aabaab".
- Delete the first 3 letters ("aab") since the next 3 letters are equal. Now, s = "aab".
- Delete the first letter ("a") since the next letter is equal. Now, s = "ab".
- Delete all the letters.
We used 4 operations so return 4. It can be proven that 4 is the maximum number of operations needed.
```

Example 3:

```
Input: s = "aaaaa"
Output: 5
Explanation: In each operation, we can delete the first letter of s.
```

Constraints:

- `1 <= s.length <= 4000`
- `s` consists only of lowercase English letters.

JavaScript

```
const deleteString = (s) => {
    let n = s.length, dp = Array(n).fill(0);
    for (let i = n - 1; ~i; i--) {
        let suf = s.slice(i), z = z_function(suf);
        dp[i] = 1;
        for (let j = 1; j <= suf.length / 2; j++) {
            if (z[j] >= j) {
                dp[i] = Math.max(dp[i], dp[i + j] + 1);
            }
        }
    }
    return dp[0];
};

const z_function = (s) => {
    let n = s.length, l = 0, r = 0, z = Array(n).fill(0);
    for (let i = 1; i < n; i++) {
        if (i <= r) z[i] = Math.min(r - i + 1, z[i - l]);
        while (i + z[i] < n && s[z[i]] == s[i + z[i]]) z[i]++;
        if (i + z[i] - 1 > r) {
            l = i;
            r = i + z[i] - 1;
        }
    }
    return z;
};
```

□ **Custom Testcase** Use Example Testcases

Run Submit

Submission Result: Accepted (/submissions/detail/813296051/) More Details (/submissions/detail/813296051/)

Share your acceptance!