Problem 3003: Maximize the Number of Partitions After Operations

Problem Information

Difficulty: Hard
Acceptance Rate: 53.77%
Paid Only: No
Tags: String, Dynamic Programming, Bit Manipulation, Bitmask

Problem Description

You are given a string `s` and an integer `k`.

First, you are allowed to change **at most** **one** index in `s` to another lowercase English letter.

After that, do the following partitioning operation until `s` is **empty** :

* Choose the **longest** **prefix** of `s` containing at most `k` **distinct** characters. * **Delete** the prefix from `s` and increase the number of partitions by one. The remaining characters (if any) in `s` maintain their initial order.

Return an integer denoting the **maximum** number of resulting partitions after the operations by optimally choosing at most one index to change.

Example 1:

Input: s = "accca", k = 2

Output: 3

Explanation:

The optimal way is to change `s[2]` to something other than a and c, for example, b. then it becomes `"acbca"`.

Then we perform the operations:

1. The longest prefix containing at most 2 distinct characters is `"ac"`, we remove it and `s` becomes `"bca"`. 2. Now The longest prefix containing at most 2 distinct characters is `"bc"`, so we remove it and `s` becomes `"a"`. 3. Finally, we remove `"a"` and `s` becomes empty, so the procedure ends.

Doing the operations, the string is divided into 3 partitions, so the answer is 3.

Example 2:

Input: s = "aabaab", k = 3

Output: 1

Explanation:

Initially `s` contains 2 distinct characters, so whichever character we change, it will contain at most 3 distinct characters, so the longest prefix with at most 3 distinct characters would always be all of it, therefore the answer is 1.

Example 3:

Input: s = "xxyz", k = 1

Output: 4

Explanation:

The optimal way is to change `s[0]` or `s[1]` to something other than characters in `s`, for example, to change `s[0]` to `w`.

Then `s` becomes `"wxyz"`, which consists of 4 distinct characters, so as `k` is 1, it will divide into 4 partitions.

Constraints:

* `1 <= s.length <= 104` * `s` consists only of lowercase English letters. * `1 <= k <= 26`

Code Snippets

C++:

```
class Solution {
public:
int maxPartitionsAfterOperations(string s, int k) {

}
};
```

Java:

```
class Solution {
public int maxPartitionsAfterOperations(String s, int k) {

}
}
```

Python3:

```
class Solution:
def maxPartitionsAfterOperations(self, s: str, k: int) -> int:
```