Problem 1915: Number of Wonderful Substrings

Problem Information

Difficulty: Medium
Acceptance Rate: 66.63%
Paid Only: No
Tags: Hash Table, String, Bit Manipulation, Prefix Sum

Problem Description

A **wonderful** string is a string where **at most one** letter appears an **odd** number of times.

* For example, `"ccjjc"` and `"abab"` are wonderful, but `"ab"` is not.

Given a string `word` that consists of the first ten lowercase English letters (`'a'` through `'j'`), return _the**number of wonderful non-empty substrings** in _`word` _. If the same substring appears multiple times in_`word` _, then count**each occurrence** separately._

A **substring** is a contiguous sequence of characters in a string.

Example 1:

Input: word = "aba" **Output:** 4 **Explanation:** The four wonderful substrings are underlined below: - "_**a**_ ba" -> "a" - "a _**b**_ a" -> "b" - "ab _**a**_ " -> "a" - "_**aba**_ " -> "aba"

Example 2:

Input: word = "aabb" **Output:** 9 **Explanation:** The nine wonderful substrings are underlined below: - "**_a_** abb" -> "a" - "_**aa**_ bb" -> "aa" - "_**aab**_ b" -> "aab" - "_**aabb**_ " -> "aabb" - "a _**a**_ bb" -> "a" - "a _**abb**_ " -> "abb" - "aa _**b**_ b" -> "b" - "aa _**bb**_ " -> "bb" - "aab _**b**_ " -> "b"

Example 3:

Input: word = "he" **Output:** 2 **Explanation:** The two wonderful substrings are underlined below: - "**_h_** e" -> "h" - "h** _e_** " -> "e"

Constraints:

* `1 <= word.length <= 105` * `word` consists of lowercase English letters from `'a'` to `'j'`.

Code Snippets

C++:

```
class Solution {
public:
long long wonderfulSubstrings(string word) {

}
};
```

Java:

```
class Solution {
public long wonderfulSubstrings(String word) {

}
}
```

Python3:

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