Problem 2957: Remove Adjacent Almost-Equal Characters

Problem Information

Difficulty: Medium
Acceptance Rate: 52.56%
Paid Only: No
Tags: String, Dynamic Programming, Greedy

Problem Description

You are given a **0-indexed** string `word`.

In one operation, you can pick any index `i` of `word` and change `word[i]` to any lowercase English letter.

Return _the**minimum** number of operations needed to remove all adjacent **almost-equal** characters from_ `word`.

Two characters `a` and `b` are **almost-equal** if `a == b` or `a` and `b` are adjacent in the alphabet.

Example 1:

Input: word = "aaaaa" **Output:** 2 **Explanation:** We can change word into "a** _c_** a _**c**_ a" which does not have any adjacent almost-equal characters. It can be shown that the minimum number of operations needed to remove all adjacent almost-equal characters from word is 2.

Example 2:

Input: word = "abddez" **Output:** 2 **Explanation:** We can change word into "**_y_** bd _**o**_ ez" which does not have any adjacent almost-equal characters. It can be shown that the minimum number of operations needed to remove all adjacent almost-equal characters from word is 2.

Example 3:

Input: word = "zyxyxyz" **Output:** 3 **Explanation:** We can change word into "z _**a**_ x _**a**_ x** _a_** z" which does not have any adjacent almost-equal characters. It can be shown that the minimum number of operations needed to remove all adjacent almost-equal characters from word is 3.

Constraints:

* `1 <= word.length <= 100` * `word` consists only of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
int removeAlmostEqualCharacters(string word) {

}
};
```

Java:

```
class Solution {
public int removeAlmostEqualCharacters(String word) {

}
}
```

Python3:

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