Problem 3579: Minimum Steps to Convert String with Operations

Problem Information

Difficulty: Hard
Acceptance Rate: 41.78%
Paid Only: No
Tags: String, Dynamic Programming, Greedy

Problem Description

You are given two strings, `word1` and `word2`, of equal length. You need to transform `word1` into `word2`.

For this, divide `word1` into one or more **contiguous substrings**. For each substring `substr` you can perform the following operations:

1. **Replace:** Replace the character at any one index of `substr` with another lowercase English letter.

2. **Swap:** Swap any two characters in `substr`.

3. **Reverse Substring:** Reverse `substr`.

Each of these counts as **one** operation and each character of each substring can be used in each type of operation at most once (i.e. no single index may be involved in more than one replace, one swap, or one reverse).

Return the **minimum number of operations** required to transform `word1` into `word2`.

Example 1:

Input: word1 = "abcdf", word2 = "dacbe"

Output: 4

Explanation:

Divide `word1` into `"ab"`, `"c"`, and `"df"`. The operations are:

* For the substring `"ab"`, * Perform operation of type 3 on `"ab" -> "ba"`. * Perform operation of type 1 on `"ba" -> "da"`. * For the substring `"c"` do no operations. * For the substring `"df"`, * Perform operation of type 1 on `"df" -> "bf"`. * Perform operation of type 1 on `"bf" -> "be"`.

Example 2:

Input: word1 = "abceded", word2 = "baecfef"

Output: 4

Explanation:

Divide `word1` into `"ab"`, `"ce"`, and `"ded"`. The operations are:

* For the substring `"ab"`, * Perform operation of type 2 on `"ab" -> "ba"`. * For the substring `"ce"`, * Perform operation of type 2 on `"ce" -> "ec"`. * For the substring `"ded"`, * Perform operation of type 1 on `"ded" -> "fed"`. * Perform operation of type 1 on `"fed" -> "fef"`.

Example 3:

Input: word1 = "abcdef", word2 = "fedabc"

Output: 2

Explanation:

Divide `word1` into `"abcdef"`. The operations are:

* For the substring `"abcdef"`, * Perform operation of type 3 on `"abcdef" -> "fedcba"`. * Perform operation of type 2 on `"fedcba" -> "fedabc"`.

Constraints:

* `1 <= word1.length == word2.length <= 100` * `word1` and `word2` consist only of lowercase English letters.

Code Snippets

C++:

```
class Solution {
public:
int minOperations(string word1, string word2) {

}
};
```

Java:

```
class Solution {
public int minOperations(String word1, String word2) {

}
}
```

Python3:

```
class Solution:
def minOperations(self, word1: str, word2: str) -> int:
```