Problem 488: Zuma Game

Problem Information

Difficulty: Hard
Acceptance Rate: 30.67%
Paid Only: No
Tags: String, Dynamic Programming, Stack, Breadth-First Search, Memoization

Problem Description

You are playing a variation of the game Zuma.

In this variation of Zuma, there is a **single row** of colored balls on a board, where each ball can be colored red `'R'`, yellow `'Y'`, blue `'B'`, green `'G'`, or white `'W'`. You also have several colored balls in your hand.

Your goal is to **clear all** of the balls from the board. On each turn:

* Pick **any** ball from your hand and insert it in between two balls in the row or on either end of the row. * If there is a group of **three or more consecutive balls** of the **same color** , remove the group of balls from the board. * If this removal causes more groups of three or more of the same color to form, then continue removing each group until there are none left. * If there are no more balls on the board, then you win the game. * Repeat this process until you either win or do not have any more balls in your hand.

Given a string `board`, representing the row of balls on the board, and a string `hand`, representing the balls in your hand, return _the**minimum** number of balls you have to insert to clear all the balls from the board. If you cannot clear all the balls from the board using the balls in your hand, return _`-1`.

Example 1:

Input: board = "WRRBBW", hand = "RB" **Output:** -1 **Explanation:** It is impossible to clear all the balls. The best you can do is: - Insert 'R' so the board becomes WRR _R_ BBW. W _RRR_ BBW -> WBBW. - Insert 'B' so the board becomes WBB _B_ W. W _BBB_ W -> WW. There are still balls remaining on the board, and you are out of balls to insert.

Example 2:

Input: board = "WWRRBBWW", hand = "WRBRW" **Output:** 2 **Explanation:** To make the board empty: - Insert 'R' so the board becomes WWRR _R_ BBWW. WW _RRR_ BBWW -> WWBBWW. - Insert 'B' so the board becomes WWBB _B_ WW. WW _BBB_ WW -> _WWWW_ -> empty. 2 balls from your hand were needed to clear the board.

Example 3:

Input: board = "G", hand = "GGGGG" **Output:** 2 **Explanation:** To make the board empty: - Insert 'G' so the board becomes G _G_. - Insert 'G' so the board becomes GG _G_. _GGG_ -> empty. 2 balls from your hand were needed to clear the board.

Constraints:

* `1 <= board.length <= 16` * `1 <= hand.length <= 5` * `board` and `hand` consist of the characters `'R'`, `'Y'`, `'B'`, `'G'`, and `'W'`. * The initial row of balls on the board will **not** have any groups of three or more consecutive balls of the same color.

Code Snippets

C++:

```
class Solution {
public:
int findMinStep(string board, string hand) {

}
};
```

Java:

```
class Solution {
public int findMinStep(String board, String hand) {

}
}
```

Python3:

```
class Solution:
def findMinStep(self, board: str, hand: str) -> int:
```