72. Edit Distance

Hard 8535 96 Add to List Share

Given two strings `word1` and `word2`, return *the minimum number of operations required to convert* `word1` *to* `word2`.

You have the following three operations permitted on a word:

- Insert a character
- Delete a character
- Replace a character

Example 1:

```
Input: word1 = "horse", word2 = "ros"
Output: 3
Explanation:
horse -> rorse (replace 'h' with 'r')
rorse -> rose (remove 'r')
rose -> ros (remove 'e')
```

Example 2:

```
Input: word1 = "intention", word2 = "execution"
Output: 5
Explanation:
intention -> inention (remove 't')
inention -> enention (replace 'i' with 'e')
enention -> exention (replace 'n' with 'x')
exention -> exection (replace 'n' with 'c')
exection -> execution (insert 'u')
```

Constraints:

- `0 <= word1.length, word2.length <= 500`
- `word1` and `word2` consist of lowercase English letters.

```
int minDistance(string word1, string word2) {
    int m = word1.length(), n = word2.length();
    vector<vector<int>> dp(m+1, vector<int> (n+1, 0));

    for(int i=1; i<=m; i++) {
        dp[i][0] = i;
    }

    for(int j=1; j<=n; j++) {
        dp[0][j] = j;
    }

    for(int i=1; i<=m; i++) {
        for(int j=1; j<=n; j++) {
            if(word1[i-1] == word2[j-1]) {
                dp[i][j] = dp[i-1][j-1];
            } else {
                dp[i][j] = 1 + min(dp[i][j-1], min(dp[i-1][j-1], dp[i-1][j]));
            }
        }
    }
    return dp[m][n];
}
```

#100daysofDSA





/rvislive

Rakesh Vishwakarma