```c
int minDistance(char* word1, char* word2) {
    int** dp;
    int word1_index;
    int word2_index;
    int word1_len;
    int word2_len;


    word1_len = strlen(word1);
    word2_len = strlen(word2);


    dp = (int**)malloc(sizeof(int*) * (word1_len+1) );
    memset(dp, 0x0, sizeof(int*) * (word1_len+1));
    dp[0] = (int*)malloc(sizeof(int)*(word2_len+1)*(word1_len+1));

    for(word1_index = 0; word1_index <= word1_len; word1_index++)
    {
        if(NULL == dp[word1_index])
        {
            dp[word1_index] = dp[0] + word1_index*(word2_len+1);
        }

        dp[word1_index][0] = word1_index;
    }

    for(word2_index = 0; word2_index <= word2_len; word2_index++)
    {
        dp[0][word2_index] = word2_index;
    }

    for(word1_index = 1; word1_index <= word1_len; word1_index++)
    {
        for(word2_index = 1; word2_index <= word2_len; word2_index++)
        {


            if(word1[word1_index-1] == word2[word2_index-1])
            {
                dp[word1_index][word2_index] = dp[word1_index-1][word2_index-1];

            }else
            {
                if( dp[word1_index-1][word2_index-1] <= dp[word1_index][word2_index-1] &&
                    dp[word1_index-1][word2_index-1] <= dp[word1_index-1][word2_index]
                  )
                {
                    dp[word1_index][word2_index] = dp[word1_index-1][word2_index-1] + 1;
                }else if(   dp[word1_index-1][word2_index] <= dp[word1_index-1][word2_index-1] &&
                            dp[word1_index-1][word2_index] <= dp[word1_index][word2_index-1]
                        )
                {
                    dp[word1_index][word2_index] = dp[word1_index-1][word2_index] + 1;
                }else
                {
                    dp[word1_index][word2_index] = dp[word1_index][word2_index-1] + 1;
                }

            }


        }
    }

    return dp[word1_len][word2_len];

}
```