72. Edit Distance

题目描述:https://leetcode.com/problems/edit-distance/

> 求word1要改变多少位能到word2
> 可以插入，删除，修改任意位

解题思路:

dp求编辑距离。
dp[i][j]代表word1的前i位和word2的前j位的编辑距离。
dp[i][j] = dp[i-1][j-1] , word1[i] == word2[j] dp[i][j] = min(dp[i-1][j-1], dp[i-1][j], dp[i][j-1]) + 1, word1[i] != word2[j]

代码 无压缩:

```
class Solution {
public:
    int minDistance(string word1, string word2) {
        int len1 = word1.size(), len2 = word2.size();
        if(len1 == 0) return len2;
        if(len2 == 0) return len1;
        vector<vector<int> > f(len1+1, vector<int>(len2+1));
        for(int i = 0; i <= len1; i++) {
            f[i][0] = i;
        }
        for(int i = 0; i <= len2; i++) {
            f[0][i] = i;
        }
        for(int i = 1; i <= len1; i++) {
            for(int j = 1; j <= len2; j++) {
                if(word1[i-1] == word2[j-1]) {
                    f[i][j] = f[i-1][j-1];
                }
                else {
                    f[i][j] = min(f[i-1][j-1]+1, min(f[i-1][j]+1, f[i][j-1]+1));
                }
            }
        }
        return f[len1][len2];
    }
};
```

代码 有压缩：

```
class Solution {
public:
    int minDistance(string word1, string word2) {
        int len1 = word1.size(), len2 = word2.size();
        if(len1 == 0) return len2;
        if(len2 == 0) return len1;
        vector<int> f(len1+1);
        for(int i = 0; i <= len1; i++) {
            f[i] = i;
        }
        for(int j = 1; j <= len2; j++) {
            int pre = f[0];
            f[0] = j;
            for(int i = 1; i <= len1; i++) {
                int t = f[i];
                if(word1[i-1] == word2[j-1]) {
                    f[i] = pre;
                }
                else {
                    f[i] = min(pre+1, min(f[i-1]+1, f[i]+1));
                }
                pre = t;
            }
        }
        return f[len1];
    }
};
```