# 牛客网-华为机试练习题 52

## 题目描述

Levenshtein 距离，又称编辑距离，指的是两个字符串之间，由一个转换成另一个所需的最少编辑操作次数。许可的编辑操作包括将一个字符替换成另一个字符，插入一个字符，删除一个字符。编辑距离的算法是首先由俄国科学家Levenshtein提出的，故又叫Levenshtein Distance。

Ex：

字符串A:abcdefg

字符串B: abcdef

通过增加或是删掉字符”g”的方式达到目的。这两种方案都需要一次操作。把这个操作所需要的次数定义为两个字符串的距离。

要求：

给定任意两个字符串，写出一个算法计算它们的编辑距离。

请实现如下接口

```
/*   功能：计算两个字符串的距离
 \*   输入：  字符串A和字符串B
 \*   输出：无
 \*   返回：如果成功计算出字符串的距离，否则返回-1
 */
```

- **public**  **static**  **int**  calStringDistance (String charA, String  charB)
- {
- **return**  0;
- }

## 输入描述:

输入两个字符串

## 输出描述:

得到计算结果

示例1

输入

```
abcdefg
abcdef
```

输出

```
1
```

## 解决代码：

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.math.BigInteger;
import java.util.Scanner;

public class Main {

    public static void main(String[] args) throws IOException {
        BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
        String sr=null;
        while((sr=br.readLine())!=null){
            char a[]=sr.toCharArray();
            String b=br.readLine();
            char c[]=b.toCharArray();
            int dp[][]=new int [a.length+1][b.length()+1];
            for(int i=1;i<=a.length;i++)
                dp[i][0]=i;
            for(int i=1;i<=c.length;i++)
                dp[0][i]=i;
            for(int i=1;i<a.length+1;i++) {
                for(int j=1;j<c.length+1;j++) {
                    if(a[i-1]==c[j-1])
                        dp[i][j]=dp[i-1][j-1];
                    else
                        dp[i][j]=Math.min(dp[i-1][j]+1, Math.min(dp[i][j-1]+1, dp[i-1][j-
1]+1));

                }
            }
            System.out.println(dp[a.length][c.length]);
        }
    }

}
```