问题描述

小明正在玩一个“翻硬币”的游戏。

桌上放着排成一排的若干硬币。我们用 \* 表示正面，用 o 表示反面（是小写字母，不是零）。

比如，可能情形是：\*\*oo\*\*\*oooo

如果同时翻转左边的两个硬币，则变为：oooo\*\*\*oooo

现在小明的问题是：如果已知了初始状态和要达到的目标状态，每次只能同时翻转相邻的两个硬币,那么对特定的局面，最少要翻动多少次呢？

我们约定：把翻动相邻的两个硬币叫做一步操作，那么要求：

输入格式

两行等长的字符串，分别表示初始状态和要达到的目标状态。每行的长度<1000

输出格式

一个整数，表示最小操作步数。

样例输入1

\*\*\*\*\*\*\*\*\*\*  
o\*\*\*\*o\*\*\*\*

样例输出1

5

样例输入2

\*o\*\*o\*\*\*o\*\*\*  
\*o\*\*\*o\*\*o\*\*\*

样例输出2

1

很容易证明，一个字符串如果可以通过翻转相邻两个的字符变成另一个字符串，则这两个字符串必定有偶数(包括0)个字符不同

直接从头开始遇见不同的字符，就翻转下一个字符，不必担心最后一个字符会不同，这样翻转次数即为最小

**翻动次数就是两个1之间的下标之差。**

#include <stdio.h>

#include<iostream>

#include<cstring>

using namespace std;

int a[1010];

int main()

{

string s1,s2;

cin>>s1>>s2;

memset(a,0,sizeof(a));

int l=s1.length();

for(int i=0;i<l;i++)

if(s1[i]!=s2[i])

a[i]=1;

int ans(0),wei=-1;

for(int i=0;i<l;i++)

if(a[i]==1)

if(wei==-1)

wei=i;

else

{

ans+=i-wei;

wei=-1;

}

cout<<ans<<"\n";

return 0;

}