**Dremplay and Upside Down**

Max. Marks: 100

Dreamplay likes the strings that read the same upside down, that is, if we reverse the entire string, the string remains unchanged. in other words, palindrome strings  
Bob does not want to upset his friend, and would thus like to make some changes, to the string PP he already has, so that the string PP is liked by dreamplay. Bob is allowed to make only two kinds of changes.

1. Add a character at end.
2. Replace a character.

Moreover, since Dreamplay is waiting for the string, Bob wants to do this in minimum steps.   
Can you find minimum number of steps needed to change the string PP such that it reads same from both ends.

**Input**   
Only line of input consists of the string PP

**Output**  
Print a single integer, the minimum number of steps needed.

**Constraints**

* String P consists only of lowercase latin letters.
* 1≤length(P)≤50001≤length(P)≤5000

**SAMPLE INPUT**

axcaa

**SAMPLE OUTPUT**

1

**Explanation**

We can change the char 'x' from string "axcaa" to obtain "aacaa" which satisfies the required property.   
Thus **1** step is needed.

**Time Limit:**1.0 sec(s) for each input file.

**Memory Limit:**256 MB

**Source Limit:**1024 KB

**Marking Scheme:**Marks are awarded if any testcase passes.

**Allowed Languages:**C, C++, C++14, Clojure, C#, D, Erlang, F#, Go, Groovy, Haskell, Java, Java 8, JavaScript(Rhino), JavaScript(Node.js), Julia, Kotlin, Lisp, Lisp (SBCL), Lua, Objective-C, OCaml, Octave, Pascal, Perl, PHP, Python, Python 3, R(RScript), Racket, Ruby, Rust, Scala, Swift, Visual Basic

<https://www.hackerearth.com/challenge/competitive/july-circuits-17/algorithm/dramplay-and-upside-down-da5a5a5c/>

------------ACEPTADO------------

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication2

{

class Program

{

static void Main(string[] args)

{

//string input = "fghabcbhgf";

string input = Console.ReadLine();

//string input = "axcaa";

int min\_dist = int.MaxValue;

for (int i = 0; i < input.Length; i++)

{

string copia = input.ToString();

char[] rev = input.Substring(0, i).ToCharArray();

Array.Reverse(rev);

copia += new string(rev);

int j = 0;

int k = copia.Length - 1;

int distintos = rev.Length;

while (j < k)

{

if (copia[j] != copia[k])

{

distintos++;

}

j++;

k--;

}

min\_dist = Math.Min(min\_dist, distintos);

}

Console.WriteLine(min\_dist);

Console.ReadLine();

}

}

}