**CSString**

**Deskripsi Soal:**

Kamu memiliki seorang teman bernama whoami, yang suka sekali memberikan kamu tantangan coding yang seru. Suatu hari, whoami memberikan kamu sebuah tantangan yang bernama ‘CSString’, yang merupakan kepanjangan dari ‘Comparing Slicing String’. Tantangannya adalah whoami akan memberikan dua baris string, dimana baris pertama adalah String S yang terdiri dari kata-kata yang dipisahkan oleh spasi (perlu diperhatikan bahwa spasi nya bisa satu, dan bisa banyak), dan baris kedua adalah sebuah String T yang terdiri dari dua karakter dan tiga digit angka.

Tugas kamu adalah menghitung berapa banyak kata pada String S, kemudian mencetak dan mengecek apakah setiap kata pada String S memiliki key yang sama seperti String T (input baris kedua). Key pada kata bisa diambil melalui dua karakter pertama pada string, diikuti panjang dari kata tersebut dengan leading zero 3 digit. Apabila key pada kata sama dengan String T (baris input kedua), maka akan dicetak dengan format [Found <key>], jika tidak maka langsung mencetak kata tersebut. Setelah mencetak dan mengecek semua kata pada string S, selanjutnya akan diakhiri dengan total jumlah kata yang tidak cocok dengan string Key (baris input kedua) dan mencetak string asli nya.

Sebagai orang yang sudah mengikuti kelas PBP, kamu merasa semakin tertantang untuk bisa menyelesaikan tantangan yang diberikan oleh temanmu, whoami. Buatlah program tersebut !!!

**(\*Catatan: Leading zero dalam coding merujuk pada angka nol (0) yang ditempatkan di depan angka lainnya untuk memastikan bahwa angka tersebut memiliki panjang tertentu. Misalnya, jika Anda ingin merepresentasikan angka 5 dengan panjang tiga digit, Anda akan menuliskannya sebagai 005. “)**

**Format Input:**

Terdapat dua baris pada input. Baris pertama merupakan String S, dimana String S merupakan sekumpulan kata kata yang dipisahkan oleh spasi. Baris kedua merupakan String T, dimana String T merupakan sebuah key yang nantinya akan dicocokkan dengan key pada masing masing kata di String S.

**Format Output:**

Output akan sesuai dan sama sesuai dengan kriteria dibawah.

**START: J**, dimana J merupakan total kata yang terdapat di String S.

**<word>**, dimana <word> merupakan kata pada string S, apabila key dengan String T nya cocok, maka print dengan format [Found: <key>], jika tidak print langsung <word>

**,,,**

**END: K**, dimana K merupakan total kata yang tidak cocok dengan String T (input baris kedua)

**Original String:S**, dimana S merupakan string asli seperti input baris pertama.

**Constraints:**

1 < panjang String S <= 200

panjang String key == 5

**Sample Input 1 (Standard Input):**

Techno techno TECHNO tecHNO tEcHno

te006

**Sample Output 1 (Standard Output):**

START: 5

Techno

[Found te006]

TECHNO

[Found te006]

tEcHno

END: 3

Original String: Techno techno TECHNO tecHNO tEcHno

**Sample Input 2 (Standard Input):**

hellooo heLLO heLlO hEllo HEllooo heLlO

he005

**Sample Output 2 (Standard Output):**

START: 6

hellooo

[Found he005]

[Found he005]

hEllo

HEllooo

[Found he005]

END: 3

Original String: hellooo heLLO heLlO hEllo HEllooo heLlO

**Penjelasan Case:**

Pada contoh input pertama, baris pertama, yaitu "**Techno techno TECHNO tecHNO tEcHno**", merupakan string yang perlu diekstrak satu per satu untuk dicek. Dan baris kedua, yaitu "**te006**", merupakan string yang nantinya akan dicocokkan dengan key pada setiap kata di baris pertama.

Berikut adalah gambaran yang terjadi pada setiap kata di String S:

**Techno -> key: Te006 != te006 -> print Techno**

**techno -> key: te006 == te006 -> print [Found te006]**

**TECHNO -> key: TE006 != te006 -> print TECHNO**

**tecHNO -> key: te006 == te006 -> print [Found te006]**

**tEcHno -> key: tE006 != te006 -> print tEcHno**

*(Jangan lupa sertakan enter ‘\n’ pada setiap output)*

**CSString**

**Case Description:**

You have a friend named whoami, who loves giving you fun coding challenges. One day, whoami gives you a challenge called 'CSString', which stands for 'Comparing Slicing String'. The challenge is that whoami will give you two lines of string, where the first line is an S String consisting of words separated by spaces (note that the spaces can be one, and can be many), and the second line is a T String consisting of two characters and a three-digit number.

Your job is to count how many words are in String S, then print and check if each word in String S has the same key as String T (second line input). The key of the word can be retrieved from the first two characters of the string, followed by the length of the word with a 3-digit leading zero. If the key in the word is the same as String T (second input line), it will be printed with the format [Found <key>], otherwise it will print the word directly. After printing and checking all the words in the S string, it will end with the total number of words that do not match the Key string (second input line) and print the original string.

As someone who has attended PBP classes, you feel even more challenged to be able to complete the challenge given by your friend, whoami. Make the program!!!

**(\*Note: Leading zero in coding refers to a zero (0) placed in front of another number to ensure that the number has a certain length. For example, if you want to represent the number 5 with a three-digit length, you would write it as 005. ")**

**Input Format:**

There are two lines in the input. The first line is String S, where String S is a set of words separated by spaces. The second line is String T, where String T is a key that will be matched with the key in each word in String S.

**Output Format:**

The output will be appropriate and equal according to the criteria below.

**START: J**, where J is the total words in String S.

**<word>**, where <word> is the word in string S, if the key and String T match, then print with the format [Found: <key>], otherwise print directly <word>

**,,,**

**END: K**, where K is the total words that do not match with String T (second line input)

**Original String:S**, where S is the original string as the first line input.

**Constraints:**

1 < length of String S <= 200

length of String key == 5

**Sample Input 1 (Standard Input):**

Techno techno TECHNO tecHNO tEcHno

te006

**Sample Output 1 (Standard Output):**

START: 5

Techno

[Found te006]

TECHNO

[Found te006]

tEcHno

END: 3

Original String: Techno techno TECHNO tecHNO tEcHno

**Sample Input 2 (Standard Input):**

hellooo heLLO heLlO hEllo HEllooo heLlO

he005

**Sample Output 2 (Standard Output):**

START: 6

hellooo

[Found he005]

[Found he005]

hEllo

HEllooo

[Found he005]

END: 3

Original String: hellooo heLLO heLlO hEllo HEllooo heLlO

**Case Explanation:**

In the first input example, the first line, "**Techno techno TECHNO tecHNO tEcHno**", is a string that needs to be extracted one by one to be checked. And the second line, "**te006**", is a string that will be matched with the key on each word in the first line.

Here is the description of what happens to each word in String S:

**Techno -> key: te006 != te006 -> print Techno**

**techno -> key: te006 == te006 -> print [Found te006]**

**TECHNO -> key: TE006 != te006 -> print TECHNO**

**tecHNO -> key: te006 == te006 -> print [Found te006]**

**tEcHno -> key: tE006 != te006 -> print tEcHno**

*(Don’t forget to put enter ‘\n’ for every output)*