Exercise: Text Processing

Problems for exercise and homework for the "JS Fundamentals" Course @ SoftUni.
Submit your solutions in the SoftUni judge system at: https://judge.softuni.org/Contests/1706

1. Reveal Words

Write a function, which receives **two parameters**.

The first parameter will be a string with some words **separated by ', '**.

The second parameter will be a string that contains **templates containing '*'**.

Find the word with the **same length** as the template and **replace** it.

Example

Input	Output
'great', 'softuni is ***** place for learning new programming languages'	softuni is great place for learning new programming languages
'great, learning', 'softuni is ***** place for ******** new programming languages'	softuni is great place for learning new programming languages

2. Modern Times of #(HashTag)

The input will be a **single string.**

Find all special words **starting with #**. If the found special word does not consist only of letters, then it is invalid and should not be printed.

Finally, print out all the special words you found without the label **(#)** on a new line.

Example

Input	Output
'Nowadays everyone uses # to tag a **#special** word in **#socialMedia**'	special socialMedia
'The symbol # is known **#variously** in English-speaking **#regions** as the **#number** sign'	variously regions number

3. Extract File

Write a function that receives a single string - the path to a file (the '\' character is escaped)

Your task is to subtract the **file name** and its **extension**. (Beware of files like **template.bak.pptx,** as **template.bak** should be the file name, while **pptx** is the extension).

Example

Input	Output

`'C:\\Internal\\training-internal\\Template.pptx'`	`File name: Template`
	`File extension: pptx`

`'C:\\Projects\\Data-Structures\\LinkedList.cs'`	`File name: LinkedList`
	`File extension: cs`

4. String Substring

The input will be given as **two** separated strings (a **word** as a first parameter and a **text** as a second).

Write a function that checks given text for containing a given word. The comparison should be **case insensitive.** Once you find a match, **print** the word and **stop** the program.

If you don't find the word print: **"{word} not found!"**

Example

Input	Output

`'javascript',`
`'JavaScript is the best programming language'`
	`javascript`

`'python',`
`'JavaScript is the best programming language'`
	`python not found!`

5. Replace Repeating Chars

Write a function that receives a single string and **replace** any sequence of the **same letters** with a single corresponding letter.

Examples

Input	Output
`'aaaaabbbbbcdddeeeedssaa'`	`abcdedsa`
`'qqqwerqwecccwd'`	`qwerqwecwd`

6. Pascal-Case Splitter

You will receive a **single string**.

This string is written in **PascalCase** format. Your task here is to split this string by **every word** in it.

Print them joined by **comma** and **space.**

Examples

Input	Output
`'SplitMeIfYouCanHaHaYouCantOrYouCan'`	`Split, Me, If, You, Can, Ha, Ha, You, Cant, Or, You, Can`
`'HoldTheDoor'`	`Hold, The, Door`
`'ThisIsSoAnnoyingToDo'`	`This, Is, So, Annoying, To, Do`

7. Cut and Reverse

The input will be a **single string.**

Write a function that cuts the given string **into half** and **reverse** the **two halves.**

Print each half on a **separate line.**

Examples

Input	Output
`'tluciffiDsIsihTgnizamAoSsIsihT'`	`ThisIsDifficult` `ThisIsSoAmazing`
`'sihToDtnaCuoYteBIboJsihTtAdooGoSmI'`	`IBetYouCantDoThis` `ImSoGoodAtThisJob`

8. *Hard Words

You will receive an **array**, which holds the **string** and **another array**.

The string is a letter from a young boy who does not yet know some words and you have to help him. The letter has a few **holes**, these holes are the words unknown to the boy and you must fill them with **strings from the array** you receive at the second index.

If the **length** of the hole is **4** you have to **replace** it with **string** with the **same length** and so on…

Examples

Input

```
['Hi, grandma! I\'m so ____ to write to you. ______ the winter vacation, so
_______ things happened. My dad bought me a sled. Mom started a new job as a
__________. My brother\'s ankle is ________, and now it bothers me even more.
Every night Mom cooks ___ on your recipe because it is the most delicious. I
hope this year Santa will _____ me a robot.',
['pie', 'bring', 'glad', 'During', 'amazing', 'pharmacist', 'sprained']]
```

Output

```
Hi, grandma! I'm so glad to write to you. During the winter vacation, so
amazing things happened. My dad bought me a sled. Mom started a new job as a
pharmacist. My brother's ankle is sprained, and now it bothers me even more.
```

```
Every night Mom cooks pie on your recipe because it is the most delicious. I
hope this year Santa will bring me a robot.
```

9. *Password Generator

For this problem, you have to write a function, which generates a password depending on input information. As such, you will be given an **array** of **three strings.** The first two strings will be at least **10 characters long**, the third one will be **one word.**

Your task here is to concatenate the first two strings and replace all **vowels** in the **concatenated string** with symbols from the third string. **The first vowel** must be replaced with the **first character** from the third string, the **second vowel** with the **second character** from that string, and so on. If the third string is less than the vowels count in the newly formed string you need to start over with the **character** on **the 0 index.** When you replace all vowels **reverse** the new password and print it on the console in a format:

`'Your generated password is {password}'`

Note: All replaced vowels with the characters from the third string must be upper-case, the rest of the characters are lower-case.

Examples

Input	Output

`[`
`'ilovepizza',`
`'ihatevegetables',`
`'orange'`
`]`
	`Your generated password is`
	`sElbGtNgAvRtOhEGzzNpAvRlO`

`[`
`'easymoneyeazylife',`
`'atleasttencharacters',`
`'absolute'`
`]`
	`Your generated password is`
	`srTtcUrLhcnOttsSBltAEfTlyzULyOnSmysBA`

`[`
`'areyousureaboutthisone',`
`'notquitebutitrustyou',`
`'disturbed'`
`]`
	`Your generated password is`
	`SIytsDrtDtEbBtRUqtTnSnIsDhttDEbBRrUsTSyIrD`

10. *Letters Change Numbers

John likes Math. But he also likes the English alphabet a lot. He invented a game with numbers and letters from the English alphabet. The game is simple. You get a string consisting of a **number between two letters**. Depending on whether the letter was in front of the number or after it you would perform different mathematical operations on the number to achieve the result.

First, you start with the letter **before** the number:

- If it's **uppercase** you **divide** the number by the letter's **position** in the alphabet

- If it's **lowercase** you **multiply** the number with the letter's **position** in the alphabet

Then you move to the **letter after** the number:

- If it's **uppercase** you **subtract** its position from the resulted number
- If it's **lowercase** you **add** its position to the resulted number

But the game became too easy for John is quick. He decided to complicate it a bit by doing the same but with **multiple** strings keeping track of only the **total sum** of all results. Once he started to solve this with more strings and bigger numbers it became quite hard to do it only in his mind. So he kindly asks you to write a program that calculates the **sum of all numbers after the operations on each number have been done**.

For example, You are given the sequence "**`A12b s17G`**":

We have two strings - **"A12b"** and **"s17G"**. We do the operations on each and sum them. We start with the letter before the number on the first string. **A is Uppercase** and its position in the alphabet is **1**. So we divide the number 12 with position 1 (**12/1 = 12)**. Then we move to the letter after the number. **b is lowercase** and its position is 2. So we add 2 to the resulting number (**12+2=14)**. Similarly for the second string **s is lowercase** and its position is 19 so we multiply it with the number (**17*19 = 323)**. Then we have Uppercase G with position 7, so we subtract it from the resulted number (**323 - 7 = 316)**. Finally, we sum the 2 results and we get **14 + 316=330**.

Input

The input comes as a **text,** holding the **sequence of strings**. Strings are separated by **one** or **more white spaces**.

The input data will always be valid and in the format described. There is no need to check it explicitly.

Output

Print on the console a single number: the **total sum of all processed numbers** rounded up to **two digits** after the decimal separator.

Constraints

- The **count** of the strings will be in the range **[1 … 10].**
- The numbers between the letters will be integers in the range **[1 … 2 147 483 647].**
- Time limit: 0.3 sec. Memory limit: 16 MB.

Examples

Input	Output	Comment
`'A12b s17G'`	`330.00`	12/1=12, 12+2=14, 17*19=323, 323-7=316, **14+316=330**
`'P34562Z q2576f   H456z'`	`46015.13`	
`'a1A'`	`0.00`	