Problem 4 – Encrypted matrix

Bogi and Acho love to play with numbers, so they invented a game to encrypt a message and create a diagonal matrix with it. Your task is to write a program, which helps Acho and Bogi convert a message into a number, encrypt it and fill a matrix with it.

First you should **convert** the message into a **number**. This is done when you take the **last digit** of the **ASCII code** of each character in the message and **add** those digits **next to each other**.

For example the string **"Soft Uni"** is converted to **31262505:**
(**'S'** => ASCII(8**3**), **'o'** => ASCII(11**1**), **'f'** => ASCII(10**2**), **'t'** => ASCII(11**6**), **' '** => ASCII(3**2**),
'U' => ASCII(8**5**), **'n'** => ASCII(11**0**), **'i'** => ASCII(10**5**)).

Then you should **encrypt** the converted number **digit by digit**. The formula is the following: If the digit is **even or '0'** - you should **multiply it by itself**. If the digit is **odd** – you should **add** to its value the **neighboring digits**. If there is a missing neighboring digit, you should add 0 instead of it.

If the **result** after the encrypting of a digit **is a number with two digits, you should concatenate** the result to the new number.

For example **31262505** is encrypted to **464364705:**
3 => 3+0+1=**4**, **1** => 1+3+2=**6**, **2** => 2*2=**4**, **6** => 6*6=**36**, **2** => 2*2=**4**, **5** => 5+2+0=**7**, **0** => 0*0=**0**, **5** => 5+0+0=**5**.

Finally you should fill a square diagonal matrix with the encrypted number. The size of the matrix should be the same as the number of digits in the encrypted number. The diagonal to be filled comes from the console as a character: **'\' (backslash)** represents the **main diagonal**; **'/' (slash)** means the **anti-diagonal**.

For example **464364705** is filled in the following two matrices:

\ (main diagonal)	/ (anti-diagonal)

4 0 0 0 0 0 0 0 0
0 6 0 0 0 0 0 0 0
0 0 4 0 0 0 0 0 0
0 0 0 3 0 0 0 0 0
0 0 0 0 6 0 0 0 0
0 0 0 0 0 4 0 0 0
0 0 0 0 0 0 7 0 0
0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 5
	0 0 0 0 0 0 0 0 5
	0 0 0 0 0 0 0 0 0
	0 0 0 0 0 0 7 0 0
	0 0 0 0 0 4 0 0 0
	0 0 0 0 6 0 0 0 0
	0 0 0 3 0 0 0 0 0
	0 0 4 0 0 0 0 0 0
	0 6 0 0 0 0 0 0 0
	4 0 0 0 0 0 0 0 0

Take note of the **direction** of the number in the anti-diagonal matrix.

Input

The input data should be read from the console.

On the first input line you have a string, containing the message.

On the second input line you have the direction as character: either '****' or '**/**'.

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

Output

The output data should be printed on the console.

You must print the matrix with a single space between the elements.

Constraints

- The string length will be in the range [1-500].
- Allowed working time for your program: 1 second.
- Allowed memory: 16 MB.

Examples

Input	Output

Soft
\
	4 0 0 0 0
	0 6 0 0 0
	0 0 4 0 0
	0 0 0 3 0
	0 0 0 0 6

Input	Output

Soft Uni
/
	0 0 0 0 0 0 0 0 5
	0 0 0 0 0 0 0 0 0
	0 0 0 0 0 0 7 0 0
	0 0 0 0 0 4 0 0 0
	0 0 0 0 6 0 0 0 0
	0 0 0 3 0 0 0 0 0
	0 0 4 0 0 0 0 0 0
	0 6 0 0 0 0 0 0 0
	4 0 0 0 0 0 0 0 0

Input	Output
S /	3