## Problem 1 – 9Gag Numbers

In 9Gag we like fun. We have a lot of time and we play with fun pictures and fun stories. Recently we invented a funny way to express numbers. We use the following 9 digits:

|  |  |
| --- | --- |
| 0 | -! |
| 1 | \*\* |
| 2 | !!! |
| 3 | && |
| 4 | &- |
| 5 | !- |
| 6 | \*!!! |
| 7 | &\*! |
| 8 | !!\*\*!- |

We write the numbers as sequences digits from our 9 available digits given above. The last digit of the number (the most right one) has a value as shown in the above table. The next digit on the left has a value 9 times bigger than the shown in the above table, the next digit on the left has 9\*9 times bigger value than the shown in the table and so on. Your task is to write a program to **convert a 9Gag-style number into its corresponding decimal representation**.

### Input

The input data consists of a single string – a 9Gag-style number.

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

### Output

The output data consists of a single line holding the calculated decimal representation of the given 9Gag-style number and should be printed at the console.

### Constraints

* The input number will have between 1 and 20 digits.
* Allowed working time for your program: 0.1 seconds.
* Allowed memory: 16 MB.

### Examples

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| \*!!! | 6 |  | \*\*\*!!! | 15 |  | !!!\*\*!- | 176 | !!\*\*!--!!- | 653 |