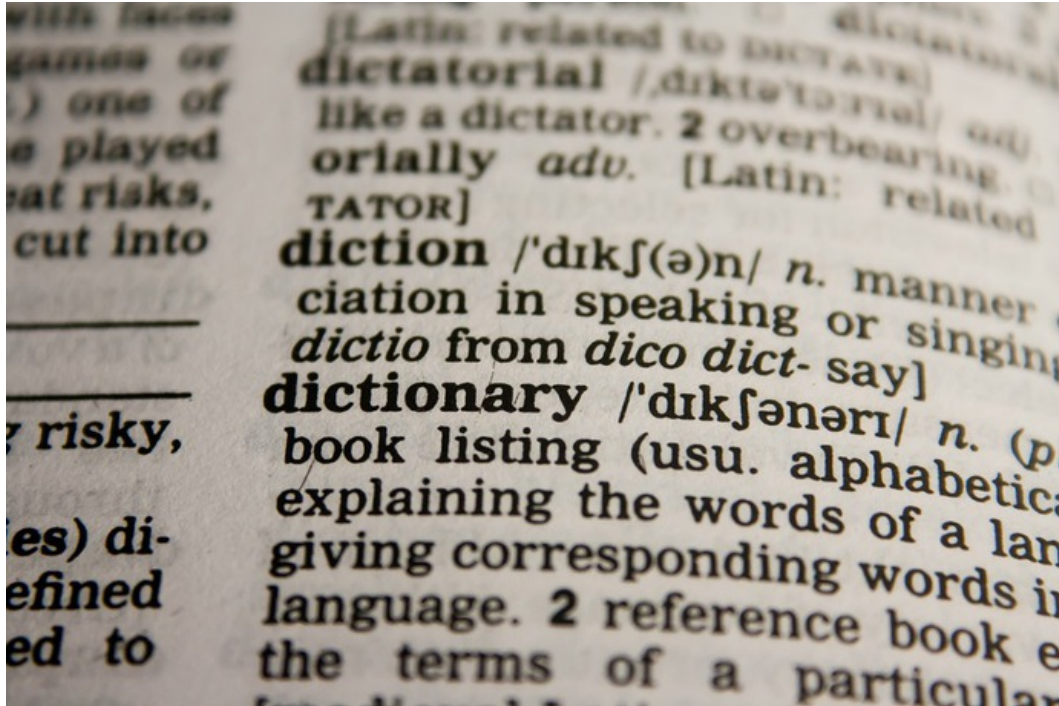


## Words, Just Words (words2)

William loves words and likes to collect them in “special” dictionaries. Today, he’s building a dictionary that contains *all the words*. Or actually: all the words that can be formed with the 26 English letters.




The size of this dictionary is, naturally, infinite. This means that it wouldn’t be practical to generate the dictionary to, say, a text file on disk, since it would require an infinite amount of space. However, William would still like to be able to identify the location in the dictionary of a given word.

The ordering rule in this special dictionary is a bit unusual: we consider shorter words to always come before longer ones. For example, the word “xy” comes before “abc”, simply because it is shorter. If two words have the same length, they are ordered in the traditional (lexicographic) way, so “xyz” > “abc”.

This means that the first 26 words in the dictionary are formed by only one character: “a”, “b”, ..., “z”. The next words are formed by only two characters, and so on.

Help William by writing a program that, given a word, calculates its zero-based index in the dictionary. That is: ‘a’ = 0, ‘b’ = 1, and so on.

 Among the attachments of this task you may find a template file `words2.*` with a sample incomplete implementation.

### Input

The first line contains the word  $W$ .

### Output

You need to write a single line with an integer: the zero-based index of  $W$  in the infinite dictionary. Since this answer can be extremely large, we only need to print it modulo 1 000 000 007.

✎ The *modulo* operation ( $a \bmod m$ ) can be written in C/C++/Python as `(a % m)` and in Pascal as `(a mod m)`. To avoid the *integer overflow* error, remember to reduce all partial results through the modulus, and not just the final result!  
Notice that if  $x < 10^9 + 7$ , then  $2x$  fits into a C/C++ `int` and Pascal `longint`.

## Constraints

- The length of the word  $W$  is between 1 and 100 000 characters.
- The word  $W$  is formed by lowercase English characters only ('a' to 'z').

## Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

- **Subtask 1** (0 points)      Examples.
- **Subtask 2** (20 points)       $N \leq 3$ .
- **Subtask 3** (20 points)      The word only uses the letter  $a$  (e.g. “aaaa”).
- **Subtask 4** (40 points)       $N \leq 1000$ .
- **Subtask 5** (20 points)      No additional limitations.

## Examples

input	output
a	0
abba	18980
verylongstring	315607945