E. E-mail Addresses

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

One of the most important products of the R1 company is a popular @r1.com mail service. The R1 mailboxes receive and send millions of emails every day.

Today, the online news thundered with terrible information. The R1 database crashed and almost no data could be saved except for one big string. The developers assume that the string contains the letters of some users of the R1 mail. Recovering letters is a tedious mostly manual work. So before you start this process, it was decided to estimate the difficulty of recovering. Namely, we need to calculate the number of different substrings of the saved string that form correct e-mail addresses.

We assume that valid addresses are only the e-mail addresses which meet the following criteria:

- the address should begin with a non-empty sequence of letters, numbers, characters '_', starting with a letter;
- then must go character '`@`';
- then must go a non-empty sequence of letters or numbers;
- then must go character '`.`';
- the address must end with a non-empty sequence of letters.

You got lucky again and the job was entrusted to you! Please note that the substring is several consecutive characters in a string. Two substrings, one consisting of the characters of the string with numbers $l_1, l_1+1, l_1+2, ..., r_1$ and the other one consisting of the characters of the string with numbers $l_2, l_2+1, l_2+2, ..., r_2$, are considered distinct if $l_1 \neq l_2$ or $r_1 \neq r_2$.

Input

The first and the only line contains the sequence of characters $s_1 s_2 ... s_n$ $(1 \leq n \leq 10^6)$ — the saved string. It is guaranteed that the given string contains only small English letters, digits and characters '`.`', '`_`', '`@`'.

Output

Print in a single line the number of substrings that are valid e-mail addresses.

Examples

input

```
gerald.agapov1991@gmail.com
```

output

```
18
```

input

```
x@x.x@x.x_e_@r1.com
```

output

```
8
```

input

```
a___@1.r
```

output

```
1
```

input

```
.asd123__..@
```

output

```
0
```

Note

In the first test case all the substrings that are correct e-mail addresses begin from one of the letters of the word agapov and end in one of the letters of the word com.

In the second test case note that the e-mail x@x.x is considered twice in the answer. Note that in this example the e-mail entries overlap inside the string.