Mixed Base Arithmetic

The Perl programming language makes it easy to implement counters that mix numeric and alphabetic characters. Lower-case letters behave as if they are a base-26 digit counting from a up to z, upper-case letters behave as if they are a base-26 digit counting from A up to Z, and decimal digits behave as usual. The fun part is that you can mix all three types of digits into a single string and incrementing still works fine, with carries propagating from one type of digit to another. For example, incrementing the string **`aB9`** gives you **`aC0`**. If a carry propagates out of the high-order digit, Perl adds a new digit of the same type as the previous high-order digit, so incrementing **`zZz9`** gives **`aaAa0`**.

Perl places some restrictions on the structure and use of a counter like this. In particular, all decimal digits must be at the right end of the counter and incrementing by one is the only available operation. In this problem, implement a Perl-like counter keeping these restrictions.

Input

Input contains several test cases, one per line. Each test case consists of a counter *c* followed by a desired increment amount *i*. The value of *c* is non-empty and may consist of up to 100 upper-case letters, lower-case letters and decimal digits. All decimal digits are to the right of any alphabet characters. The value *i* is given as a decimal integer in the range [0,1000000]. The sequence of test cases ends at the end of file.

Output

For every test case, print out a line giving the value of counter cc after it has been incremented *i* times.

Example

Standard Input	Standard Output
a 12	m
2 5	7
zZ9 1	aaA0
Zz9 1	AAa0
Zz9 6761	BAa0
aZ9 651	dN0
gb7 5017	zj4
gXrbk539 278392	gXrmc931
99 1	100

Also here:
https://open.kattis.com/problems/mixedbasearithmetic