SMART FILE NAME SORTING

You have likely tried to sort a number of files in a directory based on their names. As you have noticed, in old basic environments, the file names are sorted in the ASCII-based lexicographic order. The sorting of the alphanumeric ASCII characters is as follows:

$$0 < 1 < \ldots < 9 < A < B < \ldots < Z < a < b < \ldots < z$$

Thus, the following file names would be placed in the following order:

A, A0, A01, A02, A1, A10, A2, AA, AB, Aa, Ab, B, B0, a, a0

But, the sorting that we usually would like is the following:

a, A, a0, A0, A01, A1, A02, A2, A10, Aa, AA, Ab, AB, B, B0

Our desired sorting can be formally defined with specifying the way of comparing two file names:

1. If two file names are exactly the same, they are equal. Otherwise, they are not considered to be equal!
2. Any maximal block of consecutive digits in the file name should be considered as a single number. So, a file name is in fact a sequence of letters and numbers.
3. Two unequal file names are compared in two phases. Phase 2 is used only if the order of the two file names could not be distinguished during Phase 1.
4. Phase 1 (soft comparison): The file names are compared lexicographically based on the following rules:
    1. Numbers precede letters (a1 < aa).
    2. Numbers with lower values precede numbers with higher values (a2 < a10).
    3. Numbers with the same value are not distinguished in this phase.
    4. The letters are compared case-insensitively (in this phase only).
5. Phase 2 (exact/strict comparison): The file names are compared lexicographically based on the following rules:
    1. Numbers with the same value (but with different sequence of digits) are compared lexicographically (01 < 1 < 02 < 2 < 10).
    2. Lower case of each letter precedes its upper case form (a < A < b < B). Now, you have to write the “compare” method of our desired sorting algorithm.

Input

Each test case consists of two lines. The first string and the second string appear on the first line and the second line, respectively. Both strings are strings of at most 255 alphanumeric characters. The input terminates with "###" which should not be processed.

Output

For each test case output ‘<’, ‘=’ or ‘>’ (omit the quotes) in one line as described in the following.

- ‘<’: if the first string precedes the second one (in our desired sorting)

- ‘=’: if the two strings are exactly the same
- ‘>’: if the first string succeeds the second one (in our desired sorting)

Examples

№	stdin	stdout
1	A A A b A a 1 b Abc aBD Qwerty10 Qwerty10 100 100 010 2 010 10 A10 a2 a aa A aa 10c03 10b3 1a2b3d 01a002b3d0 ###	= < > < < = = > < > < < > <