Death to Binary?

`d.p`, `d.c`, `d.C`

The group of Absurd Calculation Maniacs has discovered a great new way how to count. Instead of using the ordinary decadic numbers, they use Fibonacci base numbers. Numbers in this base are expressed as sequences of zeros and ones similarly to the binary numbers, but the weights of bits (fits?) in the representation are not powers of two, but the elements of the Fibonacci progression ($1, 2, 3, 5, 8, \ldots$ – the progression is defined by $F_0 = 1$, $F_1 = 2$ and the recursive relation $F_n = F_{n-1} + F_{n-2}$ for $n \geq 2$).

For example $1101001_{Fib} = F_0 + F_3 + F_5 + F_6 = 1 + 5 + 13 + 21 = 40$.

You may observe that every integer can be expressed in this base, but not necessarily in a unique way – for example 40 can be also expressed as 10001001_{Fib}. However, for any integer there is a unique representation that does not contain two adjacent digits 1 – we call this representation *canonical*. For example 10001001_{Fib} is a canonical Fibonacci representation of 40.

To prove that this representation of numbers is superior to the others, ACM have decided to create a computer that will compute in Fibonacci base. Your task is to create a program that takes two numbers in Fibonacci base (not necessarily in the canonical representation) and adds them together.

Input Specification

The input consists of several instances, each of them consisting of a single line. Each line of the input contains two numbers X and Y in Fibonacci base separated by a single space. Each of the numbers has at most 40 digits. The end of input is not marked in any special way.

Output Specification

The output for each instance should be formated as follows:

The first line contains the number X in the canonical representation, possibly padded from left by spaces. The second line starts with a plus sign followed by the number Y in the canonical representation, possibly padded from left by spaces. The third line starts by two spaces followed by a string of minus signs of the same length as the result of the addition. The fourth line starts by two spaces immediately followed by the canonical representation of $X + Y$. Both X and Y are padded from left by spaces so that the least significant digits of X, Y and $X + Y$ are in the same column of the output. The output for each instance is followed by an empty line.

Sample Input

```
11101 1101
1 1
```

Output for Sample Input

```
   100101
+   10001
  -------
  1001000

   1
+  1
  --
  10
```