The standard interpretation of the binary number `1010` is $8 + 2 = 10$. An alternate way to view the sequence "`1010`" is to use Fibonacci numbers as bases instead of powers of two. For this problem, the terms of the Fibonacci sequence are:

$$1, 2, 3, 5, 8, 13, 21, \ldots$$

Where each term is the sum of the two preceding terms (note that there is only one 1 in the sequence as defined here). Using this scheme, the sequence "`1010`" could be interpreted as $1 \cdot 5 + 0 \cdot 3 + 1 \cdot 2 + 0 \cdot 1 = 7$. This representation is called a Fibinary number.

Note that there is not always a unique Fibinary representation of every number. For example the number 10 could be represented as either $8 + 2$ (`10010`) or as $5 + 3 + 2$ (`1110`). To make the Fibinary representations unique, larger Fibonacci terms must always be used whenever possible (i.e. disallow 2 adjacent `1`'s). Applying this rule to the number 10, means that 10 would be represented as 8+2 (`10010`).

Write a program that takes two valid Fibinary numbers and prints the sum in Fibinary form.

Input

The input file contains several test cases with a blank line between two consecutive.

Each test case consists in two lines with Fibinary numbers. These numbers will have at most 100 digits.

Output

For each test case, print the sum of the two input numbers in Fibinary form.

It must be a blank line between two consecutive outputs.

Sample Input

```
10010
1

10000
1000

10000
10000
```

Sample Output

```
10100

100000

100100
```