ASCII + ASCII ART

Let's do some ASCII art math and by math we mean addition. In our ASCII art we use either a dot or the lowercase letter x to represent characters. You are given an expression *x*+*y*, where *x* and *y* are positive integers. The expression is in the form of ASCII art, where all characters (the digits of *x* and *y* as well as the + sign) are represented by 7 × 5 matrices. Matrices of individual characters are concatenating together with a single column of dot characters between consecutive individual matrices. Matrices of all characters are defined as follows:

```
xxxxx ....x xxxxx xxxxx x...x xxxxx xxxxx xxxxx xxxxx xxxxx .....
x...x ....x ....x ....x x...x x.... x.... ....x x...x x...x ..x..
x...x ....x ....x ....x x...x x.... x.... ....x x...x x...x ..x..
x...x ....x xxxxx xxxxx xxxxx xxxxx xxxxx ....x xxxxx xxxxx xxxxx
x...x ....x x.... ....x ....x ....x x...x ....x x...x ....x ..x..
x...x ....x x.... ....x ....x ....x x...x ....x x...x ....x ..x..
xxxxx ....x xxxxx xxxxx ....x xxxxx xxxxx ....x xxxxx xxxxx .....
```

Your task is to compute result of the expression *x*+*y*. Both input and output should be in the ASCII form defined above.

Input

Input consists of exactly one test case in form of 7 lines. These lines describe *x*+*y* expression in the ASCII art form. Both *x* and *y* are positive integers and each of them consists of at most 9 decimal digits. All numbers are written without leading zeros.

Output

Output has exactly 7 lines containing result of the expression in ASCII art form. Do not write any leading zeros.

Example

Input

```
....x.xxxxx.xxxxx.x...x.xxxxx.xxxxx.xxxxx.......xxxxx.xxxxx.xxxxx
....x.....x.....x.x...x.x.....x.........x...x...x...x.x...x.x...x
....x.....x.....x.x...x.x.....x.........x...x...x...x.x...x.x...x
....x.xxxxx.xxxxx.xxxxx.xxxxx.xxxxx.....x.xxxxx.xxxxx.xxxxx.x...x
....x.x.........x.....x.....x.x...x.....x...x...x...x.....x.x...x
....x.x.........x.....x.....x.x...x.....x...x...x...x.....x.x...x
....x.xxxxx.xxxxx.....x.xxxxx.xxxxx.....x.......xxxxx.xxxxx.xxxxx
```

output

```
....x.xxxxx.xxxxx.xxxxx.x...x.xxxxx.xxxxx
....x.....x.....x.x.....x...x.x.........x
....x.....x.....x.x.....x...x.x.........x
....x.xxxxx.xxxxx.xxxxx.xxxxx.xxxxx.....x
....x.x.........x.....x.....x.....x.....x
....x.x.........x.....x.....x.....x.....x
....x.xxxxx.xxxxx.xxxxx.....x.xxxxx.....x
```