Problem A: $\textbf{Con} + \dfrac{\textbf{tin}}{\textbf{ued} + \dfrac{\textbf{Frac}}{\textbf{tions}}}$

The (simple) continued fraction representation of a real number r is an expression obtained by an iterative process of representing r as a sum of its integer part and the reciprocal of another number, then writing this other number as the sum of its integer part and another reciprocal, and so on. In other words, a continued fraction representation of r is of the form

$$r = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cdots}}}$$

where $a_0, a_1, a_2, \ldots$ are integers and $a_1, a_2, \ldots > 0$. We call the a_i-values *partial quotients*. For example, in the continued fraction representation of 5.4 the partial quotients are $a_0 = 5, a_1 = 2$, and $a_2 = 2$. This representation of a real number has several applications in theory and practice.

While irrational numbers like $\sqrt{2}$ require an infinite set of partial quotients, any rational number can be written as a continued fraction with a unique, finite set of partial quotients (where the last partial quotient is never 1 in order to preserve uniqueness). Given two rational numbers in continued fraction representation, your task is to perform the four elementary arithmetic operations on these numbers and display the result in continued fraction representation.

Input

Time Limit: 3 secs, No. of Test Cases: 500, Input File Size 13.1K

Each test case consists of three lines. The first line contains two integers n_1 and n_2, $1 \leq n_i \leq 9$ specifying the number of partial quotients of two rational numbers r_1 and r_2. The second line contains the partial quotients of r_1 and the third line contains the partial quotients of r_2. The partial quotients satisfy $|a_0| \leq 10$ and $0 < a_i \leq 10$, the last partial quotient will never be 1, and r_2 is non-zero. A line containing two 0's will terminate input.

Output

For each test case, display the case number followed by the continued fraction representation of $r_1 + r_2$, $r_1 - r_2$, $r_1 \times r_2$, and r_1 / r_2 in order, each on a separate line. Use 64-bit integers for all of your calculations (`long long` in C++ and `long` in Java).

Sample Input

```
4 3
5 1 1 2
5 2 2
0 0
```

Sample Output

```
Case 1:
11
0 5
30 4 6
1 27
```