Problem 1—Perfect Numbers

After spending time in the Total Perspective Vortex, Zaphod Beeblebrox has become fascinated with numbers that remind him of himself: perfect numbers.

A perfect number is a number, like 28, that is equal to the sum of all its factors except itself: 1+2+4+7+14 = 28! Another way to look at perfect numbers is that the sum of *all* of their factors *including* themselves is equal to twice the original number. 1+2+4+7+14+28 = 56 = 2×28.

This leads to an interesting generalization of perfect numbers. Look at 42, for instance. While society has chosen not to call 42 "perfect," per se, take a look the factor sum of 42.

1+2+3+6+7+14+21+42 = 96. Now look at the factors of 96.

1+2+3+4+6+8+12+16+24+32+48+96 = 252 = 6×42 !

We can call 42 an example of a (2,6)-perfect number; we had to find 2 factor sums (first of the number itself and then of the first factor sum) and ended up with a number 6 times the original.

In general, we call an (*r*,*k*)-perfect number a number that *r* factor sum computations result in a number *k* times the original. Regular perfect numbers (like 28) can be thought of as (1,2)-perfect numbers.

You are to write a program that computes (*r*,*k*)-perfect numbers within a given range.

INPUT SPECIFICATION. Each line of the input represents a case, and each case consists of four integers separated by one space. The integers are the values of *r*, *k*, *first*, and *last*, where *r* and *k* are defined as above, and *first* and *last* are the endpoints of the region you are to search. You may assume *first* ≤ *last* and that you are to search between them inclusively. The last input line consists of four zeroes. This line is not to be processed; it simply indicates the end of input.

OUTPUT SPECIFICATION. The output cases are to be processed in the same order in which they appear in the input. Follow the sample below. Note that the output numbers are to appear in increasing numerical order and an extra **<EOLN>** follows the case number and each output case.

SAMPLE INPUT.

```
1·2·1·1000<EOLN>
2·6·1·1000<EOLN>
2·2·1·1000<EOLN>
0·0·0·0<EOLN>
<EOF>
```

SAMPLE OUTPUT.

```
Case·1:<EOLN>
<EOLN>
The·(1,2)-
perfect·numbers·between·1·and·1000·are:
<EOLN>
6<EOLN>
28<EOLN>
496<EOLN>
<EOLN>
Case·2:<EOLN>
<EOLN>
The·(2,6)-
perfect·numbers·between·1·and·1000·are:
<EOLN>
42<EOLN>
84<EOLN>
160<EOLN>
336<EOLN>
<EOLN>
Case·3:<EOLN>
<EOLN>
The·(2,2)-
perfect·numbers·between·1·and·1000·are:
<EOLN>
2<EOLN>
4<EOLN>
16<EOLN>
64<EOLN>
<EOLN>
<EOF>
```