Problem C Input: c.in Output: console

This Can't Go On Forever

"A long time ago in a galaxy far, far away…", so long ago, in fact, that the Empire did not exist and there were planets without space travel. In a far corner was a world in which the predominant pet, called *tayes*, reproduced by budding. Once a *taye* had separated from its parent, it took one time unit to mature to the point of breeding, which, coincidentally, was the length of time for a new bud to grow and separate from its parent. The *tayes* are extremely long-lived, so that it became important to investigate how many someone might have, assuming that at time zero the person did not have a *taye*, and at time one had acquired an immature one, freshly budded from the mature *taye* belonging to a friend.

This ends up giving the following recurrence: $T(0) = 0$, $T(1) = 1$, $T(n) = T(n–1) + T(n–2)$, *for* $n > 1$.

Computing at the time involved unsigned binary numbers with 24 bits. That motivated a particularly curious inhabitant, Leon, to investigate the series generated by that recurrence, but constrained by a modulus — and not just the modulo 2^{24} forced by computing hardware. So the question becomes how long a series of number is generated by this recurrence, subject to arithmetic with a particular modulus, before it begins repeating. Here are the first few series:

Modulus	Front end of the series under that modulus	Length of the Period
2	**0 1 1** 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1	3
3	**0 1 1 2 0 2 2 1** 0 1 1 2 0 2 2 1 0 1 1 2 0 2 2	8
4	**0 1 1 2 3 1** 0 1 1 2 3 1 0 1 1 2 3 1 0 1 1 2 3	6
5	**0 1 1 2 3 0 3 3 1 4 0 4 4 3 2 0 2 2 4 1** 0 1 1	20

The problem is to determine the length of the period for each modulus given in the input file and report it.

Input

The input file contains an indeterminate number of lines, each containing a single integer *mod* guaranteed to be in the range $2 \le mod \le 16777216$ (2^{24}). The final line contains a 0 as end of data and should not be processed.

Output

For each modulus in the input file, print the modulus, one blank, and then the size of the smallest period of these *T* numbers under that modulus.

Sample Input	Sample Output
2	2 3
3	3 8
4	4 6
5	5 20
6	6 24
12345678	12345678 700512
16777216	16777216 25165824
0