4713 Elias Omega Coding

Elias code can be used to efficiently encode positive integers when there is no prior information about the cardinality of the integers to be encoded and it is known that the probability of getting a large integer is smaller than or equal to the probability of getting a small integer. **For practical reasons, however, in this contest we do pose a limit on the size of the input integers. For the same reason we restrict the input integer to be greater than 1.**

Elias developed three variants of the code: the Elias gamma, Elias delta, and Elias omega coding methods. This problem presents the Elias gamma and Elias omega methods and calls for implementing the Elias omega code. The following is a **background**, **definition**, and **illustration** of the problem.

Background

Suppose that Alice wants to transmit a positive integer n to Bob through a binary channel and let $\beta(n)$ stand for the binary representation of n. If Bob knows $|\beta(n)|$ (the number of bits required for the binary representation of n) in advance, then Alice should use $\beta(n)$ for the transmission. On the other hand, if Bob does not have this information, then Alice can first send $|\beta(n)|$, using efficient and distinguishable encoding, then she can send the actual beta code $F_1 = \beta(n)$. The end result is a two field code $< F_1, F_2 >$.

Elias code and its variants differ in the way they encode these two pieces of information (F_1 and F_2). The main difference between variants lies in the representation of F_1. This may imply modifications in the representation of F_1. In addition, some of the variants apply repetition or recursion to the representation of F_1. **We use a specific variant specified below**.

Definition

Formally, in Elias gamma coding, a positive integer n is encoded using two concatenated bit fields. The first field, the **prefix**, contains $\lfloor \log_2 n \rfloor$ bits of 0 ($\lfloor x \rfloor$ is the floor of x). The second field, the **postfix**, is the actual binary representation of n using $\lfloor \log_2 n \rfloor + 1$ bits. For example, the binary representation of the decimal number 9 is 1001. Under Elias coding 9 is encoded as 0001001. The first three leading zeros denote that four bits are required for the binary representation of 9. The next four bits contain the binary representation of 9. Elias delta code applies the gamma code to the prefix($\lfloor \log_2 n \rfloor$) of the gamma code and *Elias omega code* applies a recursion over the prefix Elias gamma representation of $\lfloor \log_2 n \rfloor$.

Illustration (detailed example)

To further illustrate the Elias omega code, consider the integer 536870907. The binary representation of this integer is '1 1111 1111 1111 1111 1111 1111 1011' which is required 29 bits. Hence, in the first step it is encoded as follows:

$$< F_1, F_2 >=< 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000, 1\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1011 >$$

where blanks, dots, commas, and brackets, are inserted for readability.

To emphasize, for this contest the recursion stops when the first field of a recursive stage contains one 0.

Looking at all the bits generated by all the steps and using $\beta(n)$ to denote the binary representation of n, we have:

a. <28 0's, $\beta(536870907)$ > =

$< 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000, 1\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1011 >$

b. $<<4\ 0\text{'s}, \beta(28)>, \beta(536870907)> =$
$<<0000, 1\ 1100>, 1\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1011>$

c. $<<<2\ 0\text{'s}, \beta(4)>, \beta(28)>, \beta(536870907)> =$
$<<<00, 100>, 1\ 1100>, 1\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1011>$

d. $<<<<\text{one } 0, \beta(2)>, \beta(4)>, \beta(28)>, \beta(536870907)> =$
$<<<<0, 10>, 100>, 1\ 1100>, 1\ 1111\ 1111\ 1111\ 1111\ 1111\ 1111\ 1011>$

Taking away all the blanks, dots, commas, and brackets we get that the Elias omega code of 536870907 is: '01010011100111111111111111111111111111011'. This code is distinguishable (uniquely decodable). It can be decoded in a unique way and yield back the number 536870907.

Problem statement

Given some positive integers in the range of $2 \ldots 2 \times 10^9$, you are to write a program to produce the Elias omega codes for these integers.

Input

The input contains positive integers each in a separate line. Each integer is in between 2 and 2,000,000,000, inclusive. The end of input is indicated by a zero. The input can contain up to one hundred lines.

Output

The output consists of lines each corresponding to an input integer except the last zero. Each line contains the Elias omega code of each input integer. The output should not contain any blanks or blank lines.

Sample Input

```
2
510
7
120000
536870905
49
5
0
```

Sample Output

```
010
0111000111111110
010111
01010010000111010100110000000
0101001110011111111111111111111111111001
010101110001
010101
```