The newborn Macaw baby is learning computer now. But as the Macaw dad cannot afford much so he has bought a computer that supports only floating-point data type (No support for integers). But as birds dont give exams (no CGPA calculation), dont calculate probability (Ignoring problem-setter birds) and dont intentionally share food so they rarely do floating-point calculations. Luckily, floating-point numbers can hold many integer values perfectly. Now your job is to convince the Macaw dad how good a floating-point number is in holding integer values. For this problem floating-point numbers (Similar to real world no doubt) are stored in the following way:

Suppose there is a binary number 1011_2, then to store it in computer as a floating-point number, it is divided into two parts,(a) **mantissa** (b) **exponent**. If value of mantissa part is mv, then $\frac{1}{2} \leq mv < 1$. So it is stored as, $0.1011 * 2^4$ (Assuming there is 4 bits of storage for mantissa). In other words the four bits allocated for mantissa contains 1011 and the exponent part contains the value 4 (In binary of course). As the most significant bit of mantissa is always 1 so it is not even stored at all. So for storing 1011_2 the mantissa part contains 011 (Ignoring the most siginificantbit 1) and the exponent part contains 100 (Binary equivalent of 4). Given the number of bits allocated for mantissa and exponent, your job is to find out how many different **positive integer values** can this floating-point number represent. In real life in floating-point number a bit is used for sign of number and another bit is used for sign of exponent. Those two bits are not needed for this problem (As negative integers are not counted and negative exponent never makes a positive integer value)

Input

The input file contains at most 1000 lines of input. Each line contains two integers m ($1 \leq m \leq 60$) and e ($1 \leq e \leq 60$) which denotes the number of bits allocated for mantissa and exponent respectively.

A line containing two zeroes terminates the input. This line should not be processed.

Output

For each test case produce one line of output. This line contains an integer which denotes how many different positive integer values this floating-point number can represent. You can assume that the input values will be such that this integer will not exceed $9 * 10^{18}$.

Illustration of the first sample input:

A floating-point number that has two bits for mantissa and two bits for exponent can represent the following values (Not less than 1) **1.000000**, 1.250000, 1.500000, 1.750000, **2.000000**, 2.500000, **3.000000**, 3.500000, **4.000000**, **5.000000**, **6.000000** and **7.000000**. Of these values only 7 values are integers.

Sample Input

```
2 2
3 4
4 3
0 0
```

Sample Output

```
7
103
63
```