Transforming Bowsons

locked

Physicists have discovered a new particle known as *Bowson*. As you all probably guessed, a Bowson contains a variable number of even smaller paricles called *Bows*.

A Bowson can contain at most 100,000 Bows.

Physicists have found 5 different operations that can be performed on a Bowson. Each of these operations alters the number of Bows the Bowson has. For a Bowson with *M* Bows,

Operation 1: Operation 1 halves the number of Bows the boson has. The resulting number of Bows is M/2. For obvious reasons, this operation can only be performed if M is even.

Operation 2: Sqaures the number of Bows. Resulting number of Bows is M x M.

Operation 3: Multiplies the number of Bows by 3 and adds 1. Resulting number of Bows is 3M + 1.

Operation 4: Make the number of Bows equal to the Factorial of the current number of Bows. (result = M!)

Operation 5: Resulting number of Bows becomes the sum of M's digits in binary form. (ex: 197 => 4 since 11000101 has 4 1's)

Physicists need to transform a Bowson containing **X** bows into a Bowson containing **Y** bows by doing a sequence of operations on it.

Performing these operations is expensive. So we need to minimize the number of operations performed.

Given X and Y, find the minimum number of operations needed.

Number of Bows in the Bowson cannot exceed 100,000 at any given time. Output -1 if it's impossible.

Input Format

First line contains the integer X. Second line contains the integer Y.

Constraints

0 < X, Y <= 100,000

Limits

- Time Limit: 1s
- Memory Limit: 256MB

Output Format

First line should contain a Single integer, the minimum number of operations needed.

Sample Input 0

```
54
10
```

Sample Output 0

```
4
```

Explanation 0

A possible minimal sequence is, 54 => 163 => 26569 => 79708 => 10

54x3+1 = 163 163x163 = 26569 26569x3+1 = 79708 79708 = 10011011101011100 in binary; number of 1 bits = 10