

Water Calculator (water)

William has recently built a *Water Calculator*TM, which mimics the operation of a computer through water pipes, sinks and siphons. Due to the rudimentary nature of its circuits, the *Water Calculator*TM is able to store a single number up to a billion and perform only a handful of atomic operations:

- add or subtract 1;
- multiply by 2;
- divide by a power of 3 (which divides *exactly* the stored number).

These few operations are anyway more than sufficient for William's basic mathematical needs.

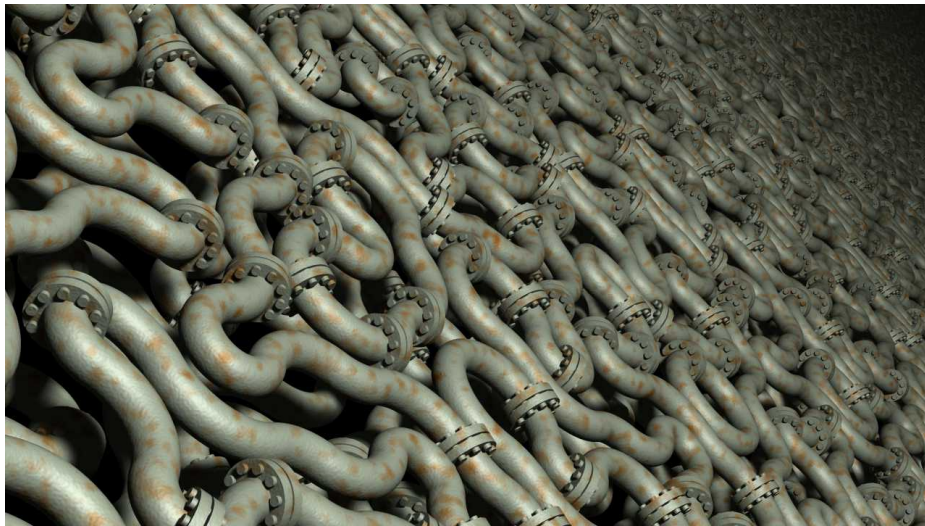



Figure 1: A small section of the *Water Calculator*TM.

In this moment, the *Water Calculator*TM is holding number N . Since William needs to leave his home for a while, it is crucial to empty the calculator in order to avoid malicious rusting! Calculate the minimum number of atomic operations necessary for William to completely empty the calculator.

 Among the attachments of this task you may find a template file `water.*` with a sample incomplete implementation.

Input

The first and only line contains the only integer N .

Output

You need to write a single line with an integer: the minimum number of atomic operations needed to empty the *Water Calculator*TM.



Constraints

- $1 \leq N \leq 10\,000\,000$.
- Operations that would increase the number stored over 10^9 are not allowed.

Scoring

Your program will be tested against several test cases grouped in subtasks. In order to obtain the score of a subtask, your program needs to correctly solve all of its test cases.

- **Subtask 1** [5 points]: Examples.
- **Subtask 2** [35 points]: $N \leq 10$.
- **Subtask 3** [30 points]: $N \leq 1000$.
- **Subtask 4** [30 points]: No additional limitations.

Examples

input.txt	output.txt
13	4
81	2

Explanation

In the **first sample case**, a possible sequence of operations is the following:

$$13 \rightarrow 26 \rightarrow 27 \rightarrow 1 \rightarrow 0$$

In the **second sample case**, a single operation brings 81 to 1 and a second one to 0.