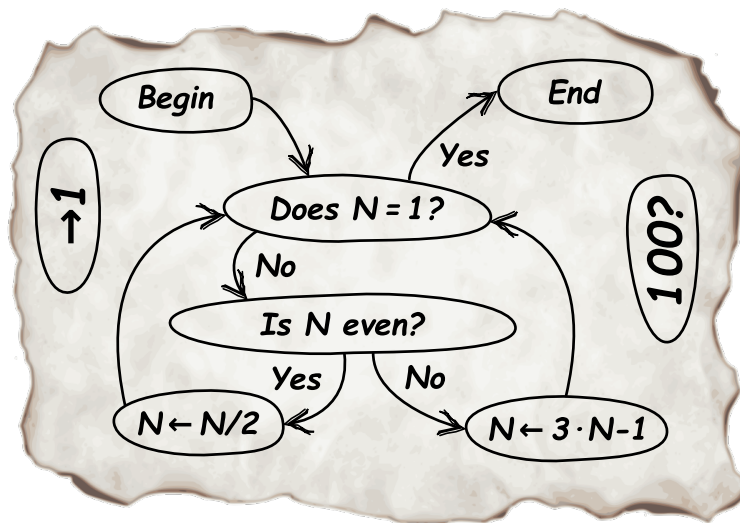


1. Ancient Code (kood)

1 sec / 3 sec

20 points

Historians found an old manuscript that seems to describe an algorithm:



The historians guess that this was used for fortune telling and it was important whether N reaches the value 1.

Write a program that gets the initial value of N and checks whether the algorithm reaches the value 1, evaluating the condition “Does $N = 1$?” no more than 100 times.

Input. The input contains one integer N_1 ($1 \leq N_1 \leq 1\,000\,000$), the initial value of N .

Output. Output one integer, the number of the iteration on which the condition $N = 1$ holds, or -1 if this does not happen in 100 iterations.

Remark. You may assume that the values of N do not exceed the range of 32-bit signed integers during the 100 iterations in any of the test cases.

Example.	Input	Output
	4	3

N gets the values 4, 2, 1. Thus the condition $N = 1$ holds on the third iteration.

Example.	Input	Output
	5	-1

N gets the values 5, 14, 7, 20, 10, 5, $\dots$, which repeat in a cycle and thus the condition $N = 1$ will never hold.

Grading. In this task, each test case is graded separately. The test cases are divided into groups where the following additional conditions hold:

- (0 points) The examples from the task statement. No points will be awarded for solving these, but the grading results will show whether your program works correctly on the evaluation server.
- (10 points) N will reach the value 1 in 100 or fewer iterations (that is, the correct answer is not -1).
- (10 points) No additional constraints.