## Problem B. pancakes

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

| | |
|---|---|
| small<br>2 points<br>*2 minute timeout* | The contest is finished. |
| large<br>11 points<br>*10 minute timeout* | The contest is finished. |

Problem

# Pancakes

At the Infinite House of Pancakes, **D** hungry diners are seated around a circular table. They are numbered from 0 at the top of the table and clockwise around to **D**-1. Each diner likes a particular type of pancake: the type with the same number as the diner.

You are a pancake server with a stack of pancakes. You are currently at the top of the table (just about to serve diner 0), and you will walk around the table clockwise, perhaps multiple times. Every time you pass by a diner, if the pancake at the top of the stack is the diner's preferred kind, you will serve that pancake. You will keep serving that diner until the top of the stack no longer matches the diner's preference; then you will move on to the next diner, and so on.

Each time you complete a revolution, after passing by diner **D**-1 and before reaching diner 0 again, you check to see if the stack is empty. If so, you are done. Otherwise, you continue clockwise around the table for another revolution, serving the diners.

For example, suppose there are 4 diners and 4 pancakes in the stack 3, 1, 2, 0 in order from top to bottom. You start your first revolution by skipping diners 0, 1 and 2 to serve diner 3 the top pancake, getting the stack down to 1, 2, 0. After that, you go back to diner 0 (starting a second revolution), skipping it to get to diner 1 and giving her pancake 1, getting the stack down to 2, 0. Since she doesn't like pancake 2, you go to diner 2, which gets it and you have only a single pancake 0 left. To serve it, you go through diner 3 and back to 0 (starting a third revolution), who gets the final pancake, and your job is done. You required 3 revolutions in total for this particular arrangement, even though you didn't need to finish the last one completely.

Note that there may be some diners who do not receive any pancakes.

Given the initial stack of pancakes, how many revolutions will you make around the table?

## Input

The input library is called "pancakes"; see the sample inputs below for examples in your language. It defines three methods:

- **GetStackSize()**:
  - Takes no argument.
  - Returns a 64-bit integer: the initial number of pancakes in the stack.
  - Expect each call to take 0.8 microseconds.
- **GetNumDiners()**:
  - Takes no argument.
  - Returns a 64-bit integer: the number **D** of diners seated around the table.
  - Expect each call to take 0.8 microseconds.
- **GetStackItem(i)**:
  - Takes a 64-bit number in the range $0 \le i <$ GetStackSize().
  - Returns a 64-bit integer: the i-th pancake type in the stack, where i = 0 corresponds to the first one we plan to serve.
  - Expect each call to take 0.8 microseconds.

## Output

Output a single integer: the number of revolutions we make around the table.

## Limits

Time limit: 3 seconds.
Memory limit per node: 128 MB.
Maximum number of messages a single node can send: 1000.
Maximum total size of messages a single node can send: 8 MB.
$0 \le$ GetStackItem(i) $<$ GetNumDiners().

Small dataset

Number of nodes: 10.
$1 \le$ GetStackSize() $\le 10^5$.
$3 \le$ GetNumDiners() $\le 10^6$.

Large dataset

Number of nodes: 100.
$1 \le$ GetStackSize() $\le 10^8$.
$3 \le$ GetNumDiners() $\le 10^9$.

Sample

| Input | Output |
| --- | --- |
| See input files below. | For sample input 1:<br>3<br>For sample input 2:<br>1<br>For sample input 3:<br>4 |

The first sample case is the one explained in the statement.
Sample input libraries:
Sample input for test 1: pancakes.h [CPP] pancakes.java [Java]
Sample input for test 2: pancakes.h [CPP] pancakes.java [Java]
Sample input for test 3: pancakes.h [CPP] pancakes.java [Java]