Luhn in Scala

Readme (readme) | Test Suite (../luhn)

Luhn

Write a program that can take a number and determine whether or not it is valid per the Luhn formula.

The Luhn formula is a simple checksum formula used to validate a variety of identification numbers, such as credit card numbers and Canadian Social Insurance Numbers.

The formula verifies a number against its included check digit, which is usually appended to a partial number to generate the full number. This number must pass the following test:

- Counting from rightmost digit (which is the check digit) and moving left, double the value of every second digit.
- For any digits that thus become 10 or more, subtract 9 from the result.
  - 1111 becomes 2121.
  - 8763 becomes 7733 (from 2×6=12 → 12-9=3 and 2×8=16 → 16-9=7).
- Add all these digits together.
  - 1111 becomes 2121 sums as 2+1+2+1 to give a check digit of 6.
  - 8763 becomes 7733, and 7+7+3+3 is 20.

If the total ends in 0 (put another way, if the total modulus 10 is congruent to 0), then the number is valid according to the Luhn formula; else it is not valid. So, 1111 is not valid (as shown above, it comes out to 6), while 8763 is valid (as shown above, it comes out to 20).

Write a program that, given a number

- Can check if it is valid per the Luhn formula. This should treat, for example, "2323 2005 7766 3554" as valid.
- Can return the checksum, or the remainder from using the Luhn method.
- Can add a check digit to make the number valid per the Luhn formula and return the original number plus that digit. This should give "2323 2005 7766 3554" in response to "2323 2005 7766 355".

About Checksums

A checksum has to do with error-detection. There are a number of different ways in which a checksum could be calculated.

When transmitting data, you might also send along a checksum that says how many bytes of data are being sent. That means that when the data arrives on the other side, you can count the bytes and compare it to the checksum. If these are different, then the data has been garbled in transmission.

In the Luhn problem the final digit acts as a sanity check for all the prior digits. Running those prior digits through a particular algorithm should give you that final digit.

It doesn't actually tell you if it's a real credit card number, only that it's a plausible one. It's the same thing with the bytes that get transmitted -- you could have the right number of bytes and still have a garbled message. So checksums are a simple sanity-check, not a real in-depth verification of the authenticity of some data. It's often a cheap first pass, and can be used to quickly discard obviously invalid things.

The Scala exercises assume an SBT project scheme. The exercise solution source should be placed within the exercise directory/src/main/scala. The exercise unit tests can be found within the exercise directory/src/test/scala.

To run the tests simply run the command `sbt test` in the exercise directory.

For more detailed info about the Scala track see the help page (http://help.exercism.io/getting-started-with-scala.html).

Source

The Luhn Algorithm on Wikipedia view source (http://en.wikipedia.org/wiki/Luhn_algorithm)