Scala

Available Exercises

Accumulate (/exercises/scala/accumulate/readme)

Implement the `accumulate` operation, which, given a collection and an operation to perform on each element of the collection, returns a new collection containing the result of applying that operation to each element of the input collection.

Allergies (/exercises/scala/allergies/readme)

Write a program that, given a person's allergy score, can tell them whether or not they're allergic to a given item, and their full list of allergies.

Anagram (/exercises/scala/anagram/readme)

Write a program that, given a word and a list of possible anagrams, selects the correct sublist.

Atbash Cipher (/exercises/scala/atbash-cipher/readme)

Create an implementation of the atbash cipher, an ancient encryption system created in the Middle East.

Bank Account (/exercises/scala/bank-account/readme)

Bank accounts can be accessed in different ways at the same time.

Binary (/exercises/scala/binary/readme)

Write a program that will convert a binary number, represented as a string (e.g. '101010'), to its decimal equivalent using first principles

Binary Search Tree (/exercises/scala/binary-search-tree/readme)

Write a program that inserts numbers and searches in a binary tree.

Bob (/exercises/scala/bob/readme)

Bob is a lackadaisical teenager. In conversation, his responses are very limited.

Clock (/exercises/scala/clock/readme)

Implement a clock that handles times without dates.

Connect (/exercises/scala/connect/readme)

Compute the result for a game of Hex / Polygon

`Crypto Square (/exercises/scala/crypto-square/readme)`

Implement the classic method for composing secret messages called a square code.

`Custom Set (/exercises/scala/custom-set/readme)`

Create a custom set type.

`Difference Of Squares (/exercises/scala/difference-of-squares/readme)`

Find the difference between the sum of the squares and the square of the sums of the first N natural numbers.

`Etl (/exercises/scala/etl/readme)`

We are going to do the `Transform` step of an Extract-Transform-Load.

`Food Chain (/exercises/scala/food-chain/readme)`

Write a program that generates the lyrics of the song 'I Know an Old Lady Who Swallowed a Fly'

`Gigasecond (/exercises/scala/gigasecond/readme)`

Write a program that will calculate the date that someone turned or will celebrate their 1 Gs anniversary.

`Grade School (/exercises/scala/grade-school/readme)`

Write a small archiving program that stores students' names along with the grade that they are in.

`Grains (/exercises/scala/grains/readme)`

Write a program that calculates the number of grains of wheat on a chessboard given that the number on each square doubles.

`Hamming (/exercises/scala/hamming/readme)`

Write a program that can calculate the Hamming difference between two DNA strands.

`Hello World (/exercises/scala/hello-world/readme)`

Beta

Write a program that greets the user by name, or by saying "Hello, World!" if no name is given.

`Hexadecimal (/exercises/scala/hexadecimal/readme)`

Write a program that will convert a hexadecimal number, represented as a string (e.g. "10af8c"), to its decimal equivalent using first principles (i.e. no, you may not use built-in or external libraries to accomplish the conversion).

`House (/exercises/scala/house/readme)`

Write a program that outputs the nursery rhyme 'This is the House that Jack Built'.

`Kindergarten Garden (/exercises/scala/kindergarten-garden/readme)`

Write a program that, given a diagram, can tell you which plants each child in the kindergarten class is responsible for.

`Largest Series Product (/exercises/scala/largest-series-product/readme)`

Write a program that, when given a string of digits, can calculate the largest product for a series of consecutive digits of length n.

`Leap (/exercises/scala/leap/readme)`

Write a program that will take a year and report if it is a leap year.

`Linked List (/exercises/scala/linked-list/readme)`

Write a Singly-Linked List implementation that uses the Proxy pattern

`Luhn (/exercises/scala/luhn/readme)`

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

`Matrix (/exercises/scala/matrix/readme)`

Write a program that, given a string representing a matrix of numbers, can return the rows and columns of that matrix.

`Meetup (/exercises/scala/meetup/readme)`

Calculate the date of meetups.

`Minesweeper (/exercises/scala/minesweeper/readme)`

Write a program that adds the numbers to a minesweeper board

`Nth Prime (/exercises/scala/nth-prime/readme)`

Write a program that can tell you what the nth prime is.

`Nucleotide Count (/exercises/scala/nucleotide-count/readme)`

Given a DNA string, compute how many times each nucleotide occurs in the string.

`Ocr Numbers (/exercises/scala/ocr-numbers/readme)`

Write a program that, given a 3 x 4 grid of pipes, underscores, and spaces, can determine which number is represented, or whether it is garbled.

`Octal (/exercises/scala/octal/readme)`

Write a program that will convert a octal number, represented as a string (e.g. '1735263'), to its decimal equivalent using first principles (i.e. no, you may not use built-in ruby libraries or gems to accomplish the conversion).

`Palindrome Products (/exercises/scala/palindrome-products/readme)`

Write a program that can detect palindrome products in a given range.

`Parallel Letter Frequency (/exercises/scala/parallel-letter-frequency/readme)`

Write a program that counts the frequency of letters in texts using parallel computation.

`Pascals Triangle (/exercises/scala/pascals-triangle/readme)`

Write a program that computes Pascal's triangle up to a given number of rows.

`Phone Number (/exercises/scala/phone-number/readme)`

Write a program that cleans up user-entered phone numbers so that they can be sent SMS messages.

`Pig Latin (/exercises/scala/pig-latin/readme)`

Implement a program that translates from English to Pig Latin

`Prime Factors (/exercises/scala/prime-factors/readme)`

Compute the prime factors of a given natural number.

`Pythagorean Triplet (/exercises/scala/pythagorean-triplet/readme)`

There exists exactly one Pythagorean triplet for which a + b + c = 1000. Find the product a * b * c.

`Queen Attack (/exercises/scala/queen-attack/readme)`

Write a program that positions two queens on a chess board and indicates whether or not they are positioned so that they can attack each other.

`Raindrops (/exercises/scala/raindrops/readme)`

Write a program that converts a number to a string, the contents of which depends on the number's prime factors.

`Rna Transcription (/exercises/scala/rna-transcription/readme)`

Write a program that, given a DNA strand, returns its RNA complement (per RNA transcription).

`Robot Name (/exercises/scala/robot-name/readme)`

Write a program that manages robot factory settings.

`Robot Simulator (/exercises/scala/robot-simulator/readme)`

Write a robot simulator.

`Roman Numerals (/exercises/scala/roman-numerals/readme)`

Write a function to convert from normal numbers to Roman Numerals: e.g.

`Saddle Points (/exercises/scala/saddle-points/readme)`

Write a program that detects saddle points in a matrix.

`Say (/exercises/scala/say/readme)`

Write a program that will take a number from 0 to 999,999,999,999 and spell out that number in English.

`Scrabble Score (/exercises/scala/scrabble-score/readme)`

Write a program that, given a word, computes the scrabble score for that word.

`Secret Handshake (/exercises/scala/secret-handshake/readme)`

Write a program that will take a decimal number, and convert it to the appropriate sequence of events for a secret handshake.

`Series (/exercises/scala/series/readme)`

Write a program that will take a string of digits and give you all the possible consecutive number series of length `n` in that string.

`Sieve (/exercises/scala/sieve/readme)`

Write a program that uses the Sieve of Eratosthenes to find all the primes from 2 up to a given number.

`Simple Cipher (/exercises/scala/simple-cipher/readme)`

Implement a simple shift cipher like Caesar and a more secure substitution cipher

`Space Age (/exercises/scala/space-age/readme)`

Write a program that, given an age in seconds, calculates how old someone is in terms of a given planet's solar years.

`Sublist (/exercises/scala/sublist/readme)`

Write a function to determine if a list is a sublist of another list.

`Triangle (/exercises/scala/triangle/readme)`

Write a program that can tell you if a triangle is equilateral, isosceles, or scalene.

`Trinary (/exercises/scala/trinary/readme)`

Write a program that will convert a trinary number, represented as a string (e.g. '102012'), to its decimal equivalent using first principles.

`Word Count (/exercises/scala/word-count/readme)`

Write a program that given a phrase can count the occurrences of each word in that phrase.

`Wordy (/exercises/scala/wordy/readme)`

Write a program that takes a word problem and returns the answer as an integer.

Getting Started

Need help getting started? You can find install documentation and resources for Scala here (http://help.exercism.io/getting-started-with-scala.html).

exercism.io
(/)

About (/about) - Donate (/donate)

GitHub (https://github.com/exercism/exercism.io) Twitter (https://twitter.com/exercism_io)

Newsletter (https://tinyletter.com/exercism)

SPONSORS

(https://bugsnag.com/blog/bugsnag-loves-open-source)

(http://www.rackspace.com/) (http://www.shopify.com/)