**Rational Numbers**

In Practice Mode

Introduction

A rational number is defined as the quotient of two integers a and b, called the numerator and denominator, respectively, where b != 0.

The absolute value |r| of the rational number r = a/b is equal to |a|/|b|.

The sum of two rational numbers r1 = a1/b1 and r2 = a2/b2 is r1 + r2 = a1/b1 + a2/b2 = (a1 \* b2 + a2 \* b1) / (b1 \* b2).

The difference of two rational numbers r1 = a1/b1 and r2 = a2/b2 is r1 - r2 = a1/b1 - a2/b2 = (a1 \* b2 - a2 \* b1) / (b1 \* b2).

The product (multiplication) of two rational numbers r1 = a1/b1 and r2 = a2/b2 is r1 \* r2 = (a1 \* a2) / (b1 \* b2).

Dividing a rational number r1 = a1/b1 by another r2 = a2/b2 is r1 / r2 = (a1 \* b2) / (a2 \* b1) if a2 \* b1 is not zero.

Exponentiation of a rational number r = a/b to a non-negative integer power n is r^n = (a^n)/(b^n).

Exponentiation of a rational number r = a/b to a negative integer power n is r^n = (b^m)/(a^m), where m = |n|.

Exponentiation of a rational number r = a/b to a real (floating-point) number x is the quotient (a^x)/(b^x), which is a real number.

Exponentiation of a real number x to a rational number r = a/b is x^(a/b) = root(x^a, b), where root(p, q) is the qth root of p.

Implement the following operations:

* addition, subtraction, multiplication and division of two rational numbers,
* absolute value, exponentiation of a given rational number to an integer power, exponentiation of a given rational number to a real (floating-point) power, exponentiation of a real number to a rational number.

Your implementation of rational numbers should always be reduced to lowest terms. For example, 4/4 should reduce to 1/1, 30/60 should reduce to 1/2, 12/8 should reduce to 3/2, etc. To reduce a rational number r = a/b, divide a and b by the greatest common divisor (gcd) of a and b. So, for example, gcd(12, 8) = 4, so r = 12/8 can be reduced to (12/4)/(8/4) = 3/2.

Assume that the programming language you are using does not have an implementation of rational numbers.

Getting Started

Make sure you have read the "Guides" section of the [C track](https://exercism.io/my/tracks/c) on the Exercism site. This covers the basic information on setting up the development environment expected by the exercises.

Passing the Tests

Get the first test compiling, linking and passing by following the [three rules of test-driven development](http://butunclebob.com/ArticleS.UncleBob.TheThreeRulesOfTdd).

The included makefile can be used to create and run the tests using the test task.

make test

Create just the functions you need to satisfy any compiler errors and get the test to fail. Then write just enough code to get the test to pass. Once you've done that, move onto the next test.

As you progress through the tests, take the time to refactor your implementation for readability and expressiveness and then go on to the next test.

Try to use standard C99 facilities in preference to writing your own low-level algorithms or facilities by hand.