99 questions/31 to 41

From HaskellWiki

< 99 questions

This is part of Ninety-Nine Haskell Problems, based on Ninety-Nine Prolog Problems (https://prof.ti.bfh.ch/hew1/informatik3/prolog/p-99/) .

1 Arithmetic

2 Problem 31

(**) Determine whether a given integer number is prime.

Example:

```
* (is-prime 7)
T
```

Example in Haskell:

```
P31> isPrime 7
True
```

Solutions

3 Problem 32

(**) Determine the greatest common divisor of two positive integer numbers. Use Euclid's algorithm (http://en.wikipedia.org/wiki/Euclidean_algorithm) .

Example:

```
* (gcd 36 63)
9
```

Example in Haskell:

```
[myGCD 36 63, myGCD (-3) (-6), myGCD (-3) 6]
```

```
[9,3,3]
```

Solutions

4 Problem 33

(*) Determine whether two positive integer numbers are coprime. Two numbers are coprime if their greatest common divisor equals 1.

Example:

```
* (coprime 35 64)
T
```

Example in Haskell:

```
* coprime 35 64
True
```

Solutions

5 Problem 34

(**) Calculate Euler's totient function phi(m).

Euler's so-called totient function phi(m) is defined as the number of positive integers r (1 <= r < m) that are coprime to m.

Example: m = 10: r = 1,3,7,9; thus phi(m) = 4. Note the special case: phi(1) = 1.

Example:

```
* (totient-phi 10)
4
```

Example in Haskell:

```
* totient 10
4
```

Solutions

6 Problem 35

(**) Determine the prime factors of a given positive integer. Construct a flat list containing the prime factors in ascending order.

Example:

```
* (prime-factors 315)
(3 3 5 7)
```

Example in Haskell:

```
> primeFactors 315
[3, 3, 5, 7]
```

Solutions

7 Problem 36

(**) Determine the prime factors of a given positive integer.

Construct a list containing the prime factors and their multiplicity.

Example:

```
* (prime-factors-mult 315)
((3 2) (5 1) (7 1))
```

Example in Haskell:

```
*Main> prime_factors_mult 315
[(3,2),(5,1),(7,1)]
```

Solutions

8 Problem 37

(**) Calculate Euler's totient function phi(m) (improved).

See problem 34 for the definition of Euler's totient function. If the list of the prime factors of a number m is known in the form of problem 36 then the function phi(m) can be efficiently calculated as follows: Let ((p1 m1) (p2 m2) (p3 m3) ...) be

the list of prime factors (and their multiplicities) of a given number m. Then phi(m) can be calculated with the following formula:

```
phi(m) = (p1 - 1) * p1 ** (m1 - 1) *
         (p2 - 1) * p2 ** (m2 - 1) *
         (p3 - 1) * p3 ** (m3 - 1) * ...
```

Note that a ** b stands for the b'th power of a.

Solutions

9 Problem 38

(*) Compare the two methods of calculating Euler's totient function.

Use the solutions of problems 34 and 37 to compare the algorithms. Take the number of reductions as a measure for efficiency. Try to calculate phi(10090) as an example.

(no solution required)

10 Problem 39

(*) A list of prime numbers.

Given a range of integers by its lower and upper limit, construct a list of all prime numbers in that range.

Example in Haskell:

```
P29> primesR 10 20
[11,13,17,19]
```

Solutions

11 Problem 40

(**) Goldbach's conjecture.

Goldbach's conjecture says that every positive even number greater than 2 is the sum of two prime numbers. Example: 28 = 5 + 23. It is one of the most famous facts in number theory that has not been proved to be correct in the general case.

It has been numerically confirmed up to very large numbers (much larger than we can go with our Prolog system). Write a predicate to find the two prime numbers that sum up to a given even integer.

Example:

```
* (goldbach 28)
(5 23)
```

Example in Haskell:

```
*goldbach 28
(5, 23)
```

Solutions

12 Problem 41

(**) Given a range of integers by its lower and upper limit, print a list of all even numbers and their Goldbach composition.

In most cases, if an even number is written as the sum of two prime numbers, one of them is very small. Very rarely, the primes are both bigger than say 50. Try to find out how many such cases there are in the range 2..3000.

Example:

```
* (goldbach-list 9 20)
10 = 3 + 7
12 = 5 + 7
14 = 3 + 11
16 = 3 + 13
18 = 5 + 13
20 = 3 + 17
* (goldbach-list 1 2000 50)
992 = 73 + 919
1382 = 61 + 1321
1856 = 67 + 1789
1928 = 61 + 1867
```

Example in Haskell:

```
*Exercises> goldbachList 9 20
[(3,7),(5,7),(3,11),(3,13),(5,13),(3,17)]
*Exercises> goldbachList' 4 2000 50
[(73,919),(61,1321),(67,1789),(61,1867)]
```

Solutions

Category:

- Tutorials
