Haskell Exercises 1: Integers

Antoni Diller

26 July 2011

(1) Write a function *sotls* (sum of two largest squares) so that *sotls* x y z is the sum of the squares of the two largest integers x, y and z.

(2) Define a function *sumsq* which takes an integer n as its argument and returns the sum of the squares of the first n integers. That is to say,

$$\mathit{sumsq}\ n = 1^2 + 2^2 + 3^2 + \ldots + n^2.$$

(3) Define the factorial function *fact* which behaves as follows: *fact* $n = 1 \times 2 \times \ldots \times n$.

(4) Define a function *comb* which takes positive integers n and m and returns the number of combinations of n objects taken m at a time. That is to say,

$$\mathit{comb}\ n\ m = \frac{n!}{m! \times (n-m)!}.$$

Note that *comb* is not defined if $n < m$. When this happens your definition should return an error message.

(5) Define a function *mygcd* which takes positive integers x and y as arguments and returns the greatest common divisor of x and y as its value. Note that *mygcd* should return an error message when both of its arguments are zero.

(6) Write a program to determine whether or not a given number is prime, that is to say, has no divisors other than 1 and itself. Call your function *prime*.

(7) A perfect number is one which is equal to the sum of its divisors, excluding itself, but including 1. Thus, 6 is perfect because $6 = 1 + 2 + 3$ and each of 1, 2 and 3 divide 6. Write a function *perfect* which tests its single argument for perfection.

(8) Suppose that you have a function *coin* which is such that *coin* i is the value of the ith coin in some currency. For example, in the United Kingdom we have:

$$\begin{aligned} \textit{coin}\ 1 &= 1,\\ \textit{coin}\ 2 &= 2,\\ \textit{coin}\ 3 &= 5,\\ \textit{coin}\ 4 &= 10,\\ \textit{coin}\ 5 &= 20,\\ \textit{coin}\ 6 &= 50,\\ \textit{coin}\ 7 &= 100,\\ \textit{coin}\ 8 &= 200. \end{aligned}$$

Write a function *countways* which is such that *countways* n m returns the number of different ways to make change from an amount m using n coins (of any value).

(9) An abundant number is a natural number whose distinct proper factors have a sum exceeding that number. Thus, 12 is abundant because $1+2+3+4+6 > 12$. Write a Boolean-valued function *abundant* which tests whether or not a number is abundant.

(10) Two numbers are amicable if each is the sum of the distinct proper factors of the other. For example, 220 and 284 are amicable because the factors of 284 are 1, 2, 4, 71 and 142 and these add up to 220 and because the factors of 220 are 1, 2, 4, 5, 10, 11, 20, 22, 44, 55 and 110 and these add up to 284. Write a function *amicable* which tests whether or not two distinct numbers are amicable.

(11) The least common multiple of a set of numbers is the smallest number that is exactly divisible by all of the numbers in the set. For example, the least common multiple of 3, 5 and 10 is 30. Write a function *lcm3* which is such that *lcm3* i j k is the least common multiple of the three numbers i, j and k,