Prof. O. Nierstrasz
Boris Spasojević, Jan Kurš

Serie 3 - Haskell

NOTE Please use the provided *template.hs* file! it contains unit tests for the required functions. Feel free to add additional tests.

Exercise 1

Write a Haskell function *isPrime* which, given an integer, returns whether or not the integer is a prime number.

Exercise 2[1]

The sum of the squares of the first ten natural numbers is,

$1^2 + 2^2 + ... + 10^2 = 385$

The square of the sum of the first ten natural numbers is,

$(1 + 2 + ... + 10)^2 = 55^2 = 3025$

Hence the difference between the sum of the squares of the first ten natural numbers and the square of the sum is $3025 - 385 = 2640$.

Write a Haskell function *diffSquareOfSumAndSumOfSquares* which, given an integer *x*, finds the difference between the square of the sum and the sum of the squares for the first *x* natural numbers.

Exercise 3

In this exercise, you are going to implement a set of functions which operate on lists. Their semantics are given below.

a. Write a function `insertNode` which adds a new node to the list. The new node should be inserted before the first node with a higher value (we assume that all lists to contain numbers).

b. Write a function `deleteNodes` which deletes all nodes which satisfy a certain predicate `p`.

c. Write a function `removeDuplicates` which removes duplicates to get a list with nodes having unique values.

d. Write a function `sumNodes` which calculates the sum of all nodes of the list.

e. Write a mapping function `mapList` which applies to each node of the list a given function `f`, e.g., the `square` function, and returns a list with the resulting values.

f. Write a function `mergeLists` which merges two sorted lists to produce one list which is also sorted.

g. Use the function from f to implement a sorting function `sortList` which sorts a list in ascending order. A *Mergesort* would be adequate in this case.

[1] This exercise is an adaption of problem 6 from `http://projecteuler.net/`. You can find a ton of cool problems there.