Functional Programming / Funkcinis programavimas

Exercise set 2

Solutions to be sent until October 27th

Exercise 1. Define a function

```
average :: [Float] -> Float,
```

which for a given number list returns their average value. (If you need to convert an integer number into a float, use function `fromIntegral` .)

Exercise 2. Write a function

```
divides :: Integer -> [Integer],
```

which for any integer number returns a list of its divisors. Create two versions of such a function, one based on recursion and the other one on the *list comprehension* method (see Lecture 5).

Relying on your `divides` implementation, define a function, which checks whether a given non-negative integer is a prime number.

Exercise 3. Write a function

```
prefix :: String -> String -> Bool,
```

which for any two given strings checks whether the first one is a prefix (i.e., coincides with the beginning) of the other one.

Relying on your `prefix` implementation, define a function `substring`, checking whether one given string is a part of another one.

Exercise 4. Define a function

```
permut :: [Integer] -> [Integer] -> Bool,
```

which checks that the two given lists of integers are permutations of each other. A permutation means that the lists consist of the same elements, occuring the same number of times. **Note : your solution must not rely on list sorting!**

Exercise 5. Using the *list comprehension* method, define a function

```
capitalise :: String -> String,
```

which modifies a given string by filtering (leaving only letters) and then changing the found letters into capital ones.

Exercise 6. A shop basket can be defined as the data structure `[(String,Float)]`, storing a pairs of items and their prices. Write two functions:

- `itemTotal :: [(String,Float)] -> [(String,Float)]`, which merges repeating items, summing their prices accordingly,
- `itemDiscount :: String -> Integer -> [(String,Float)] -> [(String,Float)]`, which applies a discount (the second parameter, ranging from 0% until 100%) to a given item (the first parameter) and correspondingly modifies the prices of this item in a given shop basket (the third parameter).