Functional Programming for BDA - List 1

Marcin Michalski, DCS FFPT WUST 2019/2020

Exercise 1. Upgrade the following function

```
factorial 1 = 1
factorial n = n * factorial (n-1)
```

so that it's tail recursive.

Exercise 2. Calculate the time complexity of the following list-reversing function

```
rev [] = []
rev (x:xs) = rev xs ++ [x]
```

Implement one that has a linear time complexity.

Exercise 3. Implement a function that for a given natural n calculates the n-th member of Fibonacci sequence in the linear time.

Exercise 4. The Euler function φ is given by the formula

$$\varphi(n) = |\{k \in \mathbb{N} : \mathrm{GCD}(n,k) = 1 \ \wedge \ 0 < k < n\}|,$$

where $\mathrm{GCD}(n,k)$ is the greatest common divisor of natural numbers n and k and $|A|$ denotes a cardinality (number of elements in a finite case) of the set A. Implement

a) the Euler function;

b) a function that calculates $\sum_{k\in\{i\in\mathbb{N}:\, i|n\}} \varphi(k)$.

Exercise 5. Implement a function that for a given list xs generates a list of pairs (ys, zs) satisfying $ys + +zs = xs$.

Exercise 6. Implement a function that eliminates consecutive duplicates from a given list, e. g.

```
ecd [1,1,2,2,1] == [1,2,1]
```

Exercise 7. Implement a function that for a given list generates a list of lists of consecutive duplicates, e. g.

```
pack [1,1,2,2,1] == [[1,1],[2,2],[1]]
```

Exercise 8. Implement a function that code a given string (which is a list of characters) by counting consecutive occurrences, e. g.

```
encode "aaabbccccaa" == [(a,3),(b,2),(c,3),(a,2)]
```

Implement a function that reverses that process.

Exercise 9. Implement a function that for a given list generates a list of its all sublists, e. g.

```
powerlist [1,2] = [[],[1],[2],[1,2]]
```

Exercise 10. Implement a function that for a given list generates a list of its all permutations.