Функциональное программирование
Задание IV

Решение присылайте в виде tar.gz-,zip-архива с исходниками или ссылку на github-репозиторий.

FP1 Реализуйте функции, имеющие следующие типы:

```
id_ :: a -> a
eval :: (a -> b, a) -> b
exchange :: (a, b) -> (b, a)
compose :: (b -> c) -> (a -> b) -> a -> c
curry_ :: ((a,b) -> c) -> (a -> b -> c)
associate :: (a, (b, c)) -> ((a, b), c)
```

FP2 Реализуйте функцию поиска минимума и максимума в списке.

```
minMax :: Ord a => [a] -> Maybe (a, a)  -- (min, max)
-- minMax [1, 2, 3, 1] == Just (1, 3)
-- minMax [1] == Just (1, 1)
-- minMax [] == Nothing
```

Ответ обернут в `Maybe` — тип, определенный в `Prelude`. Для `Java` аналогом является `Optional`. Нельзя использовать функции `minimum/maximum` из модуля `Data.List`.

FP3 Реализуйте функции, находящие сумму и количество цифр(в 10-ой системе счисления) заданного целого числа.

FP4 Реализуйте функцию поиска *преобладающего* элемента списка. Элемент списка длины n называется *преобладающим*, если он встречается в нем более чем $n/2$ раз.
Найдите решение с линейным временем работы — $O(n)$.

FP5 Реализуйте функцию f, которая принимает функцию $g :: a \to a$ и положительное целое n и возвращает функцию, вычисляющую n-кратное применение переданной функции g.

$$(f\ g\ n)\ x = \underbrace{g\ \$\ g\ \$\ g\ \$\ \ldots\ \$\ g}_{n}\ x$$

```
f:: (a -> a) -> Int -> (a -> a)
 -- f (+1) 10 $ 1 == 11
 -- f (+1) 0 $ 1  ***Exception: n must be positive number
```

FP6 Релизуйте функцию, вычисляющую последнюю цифру (в 10-ой системе счисления) n-го члена последовательности Фибоначчи.

FP7 Реализуйте рекурсивную функцию, которая определяет является ли палиндромом переданная строка. Регистр при сравнении символов *не учитывается*.

```
isPalindrome :: String -> Bool
-- isPalindrome "Aba" == True
-- isPalindrome "" == True
```