# Функциональное программирование. Задание 1.

## Условия.

```haskell
-- 1. head' возвращает первый элемент непустого списка (0,5 балла)
head' :: [a] -> a
head' = undefined
-- 2. tail' возвращает список без первого элемента, для пустого - пустой (0,5)
tail' :: [a] -> [a]
tail' = undefined
-- 3. take' возвращает первые n >= 0 элементов исходного списка (0,5)
take' :: Int -> [a] -> [a]
take' = undefined
-- 4. drop' возвращает список без первых n >= 0 элементов; если n больше длины --
списка, то пустой список. (0,5)
drop' :: Int -> [a] -> [a]
drop' = undefined
-- 5. filter' возвращает список из элементов, для которых f возвращает True (0,5)
filter' :: (a -> Bool) -> [a] -> [a]
filter' f xs = undefined
-- 6. foldl' последовательно применяет функцию f к элементу списка l и значению,
полученному на предыдущем шаге, начальное значение z (0,5)
-- foldl' (+) 0 [1, 2, 3] == (((0 + 1) + 2) + 3)
-- foldl' (*) 4 [] == 4
foldl' :: (a -> b -> a) -> a -> [b] -> a
foldl' f z l = undefined
-- 7. concat' принимает на вход два списка и возвращает их конкатенацию (0,5)
-- concat' [1,2] [3] == [1,2,3]
concat' :: [a] -> [a] -> [a]
concat' = undefined
-- 8. quickSort' возвращает его отсортированный список (0,5)
quickSort' :: Ord a => [a] -> [a]
quickSort' = undefined
```

## Требования.

1. задание сдается в виде файла **Fp1.hs**;
2. запрещено использовать любые предопределенные функции (в том числе из Prelude), но разрешено определять свои собственные вспомогательные функции;
3. **дедлайн 23.11.2016**, т. е. 23 - последний день сдачи;