Курс: Функциональное программирование
Практика 7. Свёртки

Моноиды

Моноид — это множество с ассоциативной бинарной операцией над ним (`mappend`) и единицей для этой операции (`mempty`).

Эндоморфизм (стрелка из типа в него же) можно упаковать так

```
newtype Endo a = Endo { appEndo :: a -> a }
```

Эндоморфизм образует моноид относительно композиции.

▶ Напишите

```
instance Monoid (Endo a) where
 mempty = ???
 Endo f `mappend` Endo g = ???
```

▶ Определите тип функции

```
fn = mconcat $ map Endo [(+5),(*3),(^2)]
```

и вычислите значение выражения

```
appEndo fn 2
```

▶ Можно ли написать представителя моноида для типа `Maybe a`? Сколько разных вариантов можно реализовать?

Свёртки

▶ Устно вычислите значения выражений и проверьте результат в GHCi:

```
foldl (/) 480 [3,2,5,2]
foldr (/) 2 [8,12,24,4]
```

▶ Напишите реализацию следующих функций через свёртки `foldr` или `foldr1`

```
length' :: [a] -> Int
length' =

or' :: [Bool] -> Bool
or'      =

head' :: [a] -> a
head'     =

last' :: [a] -> a
last'     =

maximum' :: Ord a => [a] -> a
maximum'  =

map' :: (a -> b) -> [a] -> [b]
map' f     =

filter' :: (a -> Bool) -> [a] -> [a]
filter' p    =
```

▶ Напишите две реализации `reverse :: [a] -> [a]` — через свёртки `foldr` и `foldl`

```
reverse'  =

reverse''  =
```

▶ Используя правую свёртку, напишите функцию, конструирующую из списка строк строку, разделённую запятыми.

```
> f ["ab","cde","fgh"]
"ab,cde,fgh"
```

▶ Напишите реализацию `foldl` через `foldr`.

▶ Напишите реализацию функции `foldTree` — свёртки для двоичного дерева

```
data Tree a = Nil | Branch (Tree a) a (Tree a)  deriving (Eq, Show)
```

▶ Используя `foldTree`, напишите функцию `flattenTree :: Tree a -> [a]`. Сколько разных версий такой функции Вы можете написать?