Haskell CheatSheet

Laborator 8

type

`type` ne permite definirea unui sinonim de tip, similar cu `typedef` din C.

```
type Point = (Int, Int)

p :: Point
p = (2, 3)
```

newtype

`newtype` este similar cu data, cu diferența că ne permite crearea unui tip de date cu un singur constructor, pe baza altor tipuri de date existente.

```
newtype Celsius = MakeCelsius Float deriving Show

newtype Fahrenheit = MakeFahrenheit Float
    deriving Show

celsiusToFahrenheit :: Celsius -> Fahrenheit
celsiusToFahrenheit (MakeCelsius c) =
    MakeFahrenheit $ c * 9/5 + 32
```

data

`data` permite definirea de noi tipuri de date algebrice.

```
data PointT = PointC Double Double deriving Show
```

Tipuri enumerate.

```
data Colour = Red | Green | Blue | Black
    deriving Show
nonColour :: Colour -> Bool
nonColour Black = True
nonColour _     = False
```

Tipuri înregistrare.

```
data PointT = PointC
    { px :: Double
    , py :: Double
    } deriving Show
px (PointC x _) = x
py (PointC _ y) = y
```

Tipuri parametrizate.

```
data Maybe a = Just a | Nothing
    deriving (Show, Eq, Ord)
maybeHead :: [a] -> Maybe a
maybeHead (x : _) = Just x
maybeHead _       = Nothing
```

Tipuri recursive.

```
data List a = Void | Cons a (List a) deriving Show

data Natural = Zero | Succ Natural deriving Show
```