Laboratorul 8: ADT. Clase de Tipuri

Exercițiul 1

Se dau următoarele tipuri de date ce reprezinta puncte cu numar variabil de coordonate intregi:

```
data Punct = Pt [Int]
```

Arbori cu informația în frunze şi clasă de tipuri `ToFromArb`

```
data Arb = Vid | F Int | N Arb Arb
          deriving Show

class ToFromArb a where
      toArb :: a -> Arb
      fromArb :: Arb -> a
```

a) Să se scrie o instanță a clasei Show pentru tipul de date Punct, astfel încât lista coordonatelor sa fie afisată sub forma de tuplu.

```
-- Pt [1,2,3]
-- (1, 2, 3)

-- Pt []
-- ()
```

b) Să se scrie o instanţă a clasei ToFromArb pentru tipul de date Punct astfel incat lista coordonatelor punctului sa coincidă cu frontiera arborelui.

```
-- toArb (Pt [1,2,3])
-- N (F 1) (N (F 2) (N (F 3) Vid))
-- fromArb $ N (F 1) (N (F 2) (N (F 3) Vid)) :: Punct
--  (1,2,3)
```

Exercițiul 2

Se dă următorul tip de date reprezentând figuri geometrice.

```
data Geo a = Square a | Rectangle a a | Circle a
    deriving Show
```

Și clasa `GeoOps` în care se definesc operațiile `perimeter` și `area`.

```
class GeoOps g where
  perimeter :: (Floating a) => g a -> a
  area :: (Floating a) =>  g a -> a
```

a) Să se instanțieze clasa `GeoOps` pentru tipul de date `Geo`. Pentru valoarea `pi` există funcția cu același nume (`pi`).

```
-- ghci> pi
-- 3.1415926535897793
```

b) Să se instanțieze clasa `Eq` pentru tipul de date `Geo`, astfel încât două figuri geometrice să fie egale dacă au perimetrul egal.