Laboratorul 9: ADT. Clase de tipuri

Arbori

Se dă următorul tip de date reprezentând arbori ternari cu informația de tip întreg în rădăcină.

```
data Tree = Empty  -- arbore vid
  | Node Int Tree Tree Tree -- arbore cu valoare de tip Int in radacina
                            -- si 3 fii


-- extree :: Tree
-- extree = Node 4 (Node 5 Empty Empty Empty)
--                 (Node 3 Empty Empty (Node 1 Empty Empty Empty)) Empty
```

1. Instanțiați clasa următoare pentru tipul `Tree`.

```
class ArbInfo t where
  level :: t -> Int -- intoarce inaltimea arborelui;
                    -- consideram ca un arbore vid are inaltimea 0
  sumval :: t -> Int -- intoarce suma valorilor din arbore
  nrFrunze :: t -> Int -- intoarce nr de frunze al arborelui

-- level extree
-- 3
-- sumval extree
-- 13
-- nrFrunze extree
-- 2
```

Vectori

În continuare, vom implementa spații vectoriale. Se dă următoarea clasă a scalarilor.

```
class Scalar a where
  zero :: a
  one :: a
  adds :: a -> a -> a
  mult :: a -> a -> a
```

```
  negates :: a -> a
  recips :: a -> a
```

2. Instanțiați clasa `Scalar` folosindu-vă de tipuri primitive (hint: nu uitați, trebuie să fie corpuri comutative). Apoi, considerați clasa de mai jos a vectorilor.

```
class (Scalar a) => Vector v a where
  zerov :: v a
  onev :: v a
  addv :: v a -> v a -> v a -- adunare vector
  smult :: a -> v a -> v a  -- inmultire cu scalare
  negatev :: v a -> v a -- negare vector
```

3. Scrieți două instanțe ale clasei `Vector` pentru a reprezenta vectori bidimensionali și tridimensionali.

Extra: corespondența Curry-Howard

Urmăriți prezentarea 'Propositions as Types' ținută de Philip Wadler
https://www.youtube.com/watch?v=IOiZatlZtGU&t=2194s
(și/sau citiți lucrarea cu același nume)
https://homepages.inf.ed.ac.uk/wadler/papers/propositions-as-types/propositions-as-types.pdf
ca să aflați mai multe despre corespondența Curry-Howard.