Laboratorul 14: Recapitulare

1. Se dau următoarele:

un tip de date ce reprezintă puncte cu număr variabil de coordonate întregi:

```
data Point = Pt [Int]
             deriving Show
```

un tip de date ce reprezintă arbori binari de căutare (cu nodurile sortate):

```
data Arb = Empty | Node Int Arb Arb
           deriving Show
```

și o clasă de tipuri `ToFromArb`:

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

Scrieți o instanță a clasei `ToFromArb` pentru tipul `Point`. Inserarea în arbore se va face ținând cont de proprietatea arborelui de a fi sortat.

2. Scrieți o funcție care primește ca argumente două numere întregi și o listă de numere întregi, și construiește, folosind lista inițială, lista numerelor aflate în intervalul definit de cele două numere. Rezolvați problema în două moduri (propuneți o soluție fără monade și o soluție cu monade).

```
getFromInterval 5 7 [1..10] == [5,6,7]
```

3. Se dă tipul de date:

```
newtype ReaderWriter env a = RW {getRW :: env-> (a,String)}
```

Scrieți instanța completă a clasei `Monad` pentru tipul `ReaderWriter`, astfel încât să păstreze proprietatea de monadă, `env` fiind o memorie nemodificabilă și concatenând toate stringurile (hint: nu este nevoie să faceți instanțe și pentru clasele `Applicative` și `Functor`).