Subiecte

P1 [3 pct]

Să se transforme un șir de caractere după codificarea următoare:

- literele mici se transformă în '*'
- cifrele se dublează
- caracterele din lista `"ADT"` se elimină

Dacă vă ajută în rezolvare, puteți folosi funcțiile `isUpper`, `isLower`, `toUpper`, `toLower`, `isDigit` din pachetul `Data.Char`.

Să se rezolve problema în două moduri (o soluție fără monade și o soluție cu monade).

Exemple:
Pentru șirul `"Ana,2"` trebuie să obțineți șirul `"*,22"`.

P2 [2 pct]

Considerăm următoarele tipuri de date:

```
data Pereche a b = MyP a b deriving Show
data Lista a = MyL [a] deriving Show
```

Considerăm clasa:

```
class MyOp m where
    myFilter :: (a -> Bool) -> (b -> Bool) -> m (Pereche a b) -> m (Pereche a b)
```

Să se instanțieze clasa `MyOp` pentru tipul de date `Lista`. Mai jos aveți un exemplu care să vă ajute:

```
lp :: Lista (Pereche Int Char)
lp = MyL [MyP 97 'a', MyP 3 'b', MyP 100 'd']
myFilter (< 10) (< 'e') lp
```

P3 [1 pct]

Se dă tipul de date:

```
newtype AE a = AE {getAE :: (Either String a, String)} deriving Show
```

Să se scrie instanța completă a clasei `Monad` pentru tipul `AE`.

Puteți să vă verificați folosind următoarele instanțe pentru clasele `Functor` și `Applicative`, și următoarea expresie:

```
instance Functor AE where
    fmap f ma = f <$> ma

instance Applicative AE where
    pure = return
    mf <*> ma = do
        f <- mf
        f <$> ma

testAE :: AE Int
testAE = ma >>= f
    where
        ma = AE (Right 7, "ana are mere ")
        f x = AE (if x `mod` 2 == 0 then Right x else Left "error", "si pere!")

AE {getAE = (Left "error", "ana are mere si pere!")}
```