Funkcionális programozás / Mintatétel

1. Egészítsük ki az alábbi `fugv` és `hatv` függvényeket úgy, hogy a `fugv` függvény meghatározza `y` hatványértékeit `0, 1, ...` hatványkitevőkön. Adjuk meg a függvények típusdefinícióit, egy kezdeti kifejezést és a kiértékelés eredményét.

```
hatv y a
   | a < 0 = error "Negativ kitevo"
   | a == 0 = 1
   | mod a 2 == 0 = ...
   | otherwise = ...

fugv y a = map ...
```

2. Mit csinálnak az alábbi függvények? Adjuk meg a függvények típusdefinícióját, egy kezdeti kifejezést és a kiértékelés eredményét.

```
fugv1 st = foldl (\ a b -> a + b) 0 st
fugv2 e st = foldl (\temp a -> if a == e then True else temp) False st
```

3. Mit csinál az alábbi függvény, ha a függvény paramétere 10? Adjuk meg a függvény típusdefinícióját, egy kezdeti kifejezést és a kiértékelés eredményét.

```
fel n = take n (fugv 1)
   where
   fugv i = [x | x <- [2, 4..2*i]] ++ fugv (i+1)

A. Komplilálási hiba
B. [2,2,4,2,4,6,2,4,6,8]
C. [2,4,4,6,6,6,8,8,8,8]
D. [2,4,2,4,2,4,2,4,2,4]
```

4. Írjunk egy függvényt, amely egy bemeneti (valós szám, valós szám, string) hármas típusú listát úgy dolgoz fel, hogy kiírja a képernyőre külön sorba a stringeket, és kettőspont, space-el elválasztva minden string után feltünteti a két valós szám átlagát. A kiíratás végére, újsorba írjuk ki a "`kiiratas vege`" szöveget.

```
Példa:
> myPutStr [(8.0, 3.5, "Mari"), (5.0, 10.0, "Feri"), (4.0, 9.0, "Laci"),
(7.0, 9.7, "Zsuzsa")]

   Mari: 5.75
   Feri: 7.5
   Laci: 6.5
   Zsuzsa: 8.25
   kiiratas vege
```

5. Egészítsük ki, az alábbi `myMain` függvényt, úgy hogy a `myMain` függvény bináris kereséssel határozza meg, hogy hányadik helyen szerepel az `elem` érték a `myL` lista elemei között. Adjuk meg a típusdefiníciókat, egy kezdeti kifejezést és a kiértékelés eredményét.

```
      myMain elem myL =
```

```
        let
              prog xs k p q
                 | q < p        = -1
                 | m > k        = ...
                 | m < k        = ...
                 | otherwise    = temp
                 where
                 temp = p + (div (q - p) 2)
                 m = xs !! temp
        in prog myL elem 0 (length myL - 1)
```

6. Alkalmazva a következő típust, írjunk függvényt, amely meghatározza egy adott egyetemen található karok és azok diáklétszámának értékpár listáját.

```
data FH = FH{
   egyetem :: [Char],
   kar :: [Char],
   diakletszam :: Int
   }deriving (Show)
```

A függvény szignatúrája legyen a következő:

```
      hataroz1 :: [Char] -> [FH] -> [([Char], Int)]
```

Példa:
legyen a bemeneti lista:

```
eLista = [FH "sapientia" "human" 100, FH "univ_Iasi" "informatika" 220,
FH "sapientia" "informatika" 150, FH "sapientia" "gepesz" 90, FH
"babes_bolyai" "informatika" 200, FH "bukarest" "technikai" 220]
```

ekkor egy függvényhívás és az eredmény a fenti bemeneti listára a következő:

```
> hataroz1 "sapientia" eLista

[("human",100),("informatika",150),("gepesz",90)]
```

7. Alkalmazva a 6. feladatnál definiált típust, határozzuk meg hogy melyik egyetem karán tanul a legtöbb diák. A függvény szignatúrája legyen a következő:

```
      hataroz2 :: [FH] -> [([Char], [Char])]
```

Példa:
Egy függvényhívás és az eredmény, a 6. feladatnál definiált eLista-ra a következő:

```
> hataroz2 eLista

[("univIasi","informatika"), ("bukarest","technikai")]
```