Домашна работа № 4 по Функционално програмиране

специалност „Компютърни науки“, II курс, I поток, 2020/2021 учебна година

Решенията трябва да са готови за автоматично тестване. Важно е програмният код да бъде добре форматиран и да съдържа коментари на ключовите места. Предайте решенията на всички задачи в *един* файл с наименование *hw4_<FN>.hs*, където *<FN>* е Вашият факултетен номер.

Домашните работи се предават като изпълнение на съответното задание в курса по ФП в Moodle (https://learn.fmi.uni-sofia.bg/course/view.php?id=6485) най-късно до 23:55 ч. на 21.01.2021 г. (четвъртък).

Приятна работа и успех!

Задача 1. Нека е дефиниран тип „сметка“ `Account`, който представя вектор с три елемента `(Int,Int,Double)` – идентификатор на сметка, идентификатор на човек и баланс по сметката. Нека е дефиниран и тип „човек“ `Person`, който представя вектор с три елемента `(Int,String,String)` – идентификатор на човек, име и местоживеене (населено място).

Ще работим с „база от данни“ на банка, представена като двойка от списък от сметки и списък от хора.

а) Дефинирайте функция `getAverageBalance :: ([Account],[Person]) -> (Person -> Bool) -> Double`, която получава като аргументи база от данни и предикат `p`. Функцията трябва да връща средния баланс по всички сметки на хората, които удовлетворяват предиката `p`.

б) Дефинирайте на функционално ниво, чрез `getAverageBalance`, функция `averageBalanceOfCities :: ([Account],[Person]) -> [String] -> Double`, която получава база от данни и списък от населени места и връща като резултат средния баланс по сметките на хората с местоживеене в някое от изброените в списъка населени места.

Примери:

```
people1 = [(1,"Ivan","Sofia"),(2,"Georgi","Burgas"),
(3,"Petar","Plovdiv"),(4,"Petya","Burgas")]
```

```
accounts1 =
[(1,1,12.5),(2,1,123.2),(3,2,13.0),(4,2,50.2),(5,2,17.2),
(6,3,18.3),(7,4,19.4)]

getAverageBalance (accounts1,people1) (\ (_,_,city) -> city ==
"Burgas") -> 24.95 (24.950000000000003)

getAverageBalance (accounts1,people1) (\ (_,(n:_),_) -> n ==
'P') -> 18.85

averageBalanceOfCities (accounts1,people1)
["Sofia","Gabrovo","Stara Zagora"] -> 67.85
```

Задача 2. Нека е дефинирано следното представяне на двоично дърво:
`data BTree = Empty | Node Int BTree BTree`
Ще наричаме един възел в дървото „*интересен*“, ако неговата стойност е равна на `2^k`, където `k` е броят на преките му наследници. Дефинирайте функция `countInteresting :: BTree -> Int`, намираща броя на „*интересните*“ възли в дадено двоично дърво.

```
t1 :: BTree                                  --    16
t1 = Node 16 (Node 0 Empty Empty)            --   /  \
             (Node 4 (Node 1 Empty Empty)    --  0    4
                     (Node 0 Empty Empty))   --      / \
                                             --     1   0

t2 :: BTree                                 --    4
t2 = Node 4 (Node 0 Empty Empty)            --   / \
            (Node 2 (Node 1 Empty Empty)    --  0   2
                    Empty)                  --     /
                                            --    1

countInteresting t1 -> 2 (4=2^2,  1=2^0)
countInteresting t2 -> 3 (4=2^2,  2=2^1, 1=2^0)
```