**კურსის დასახელება:** ფუნქციონალური დაპროგრამება **სტატუსი:** ძირითადი

**ქულების განაწილება:** 1 - **5**ქ., 2 - **5** ქ., 3 - **5** ქ., 4 - **5** ქ., 5 - 1**5** ქ. =37 **ქულათა ჯამი:** **40** ქულა

**გამოცდის ხანგრძლივობა: 2 საათი.**

**ბილეთი**

1. სიის კონსტრუქტორის გამოყენებით დაწერეთ mymap სახელწოდების მაღალი რიგის საბიბლიოთეკო ფუნქციის განმარტების თქვენი ვერსია, რომელიც ახორციელებს მითითებულ მოქმედებას (ფუნქციას) სიის ყოველ ელემენტზე . განსაზღვრეთ ფუნქციის ტიპი. მოიყვანეთ გამოძახების მაგალითი.   
     
   mymap f xs =[f x | x <- xs] +5  
   mymap :: (t1 -> t) -> [t1] -> [t]  
   mymap (\x -> x \* x) [1..10]
2. რეკურსიის საშუალებით. დაწერეთ მაღალი რიგის საბიბლიოთეკო myfilter ფუნქცია, რომელიც ირჩევს სიიდან ყოველ ელემენტს, რომელიც პრედიკატს აკმაყოფილებს განსაზღვრეთ ფუნქციის ტიპი. მოიყვანეთ გამოძახების მაგალითი.   
     
   myfilter \_ []= [] +4  
   myfilter f (x:xs) = if f x then x:myfilter f xs else myfilter f xs  
   myfilter odd [1..10]  
   პასუხი?
3. განსაზღვრეთ რეკურსიულად ან სიის კონსტრუქტორის გამოყენებით ფუნქცია, რომელიც შესასვლელზე ღებულობს ნამდვილ რიცხვების სიას და ითვლის ორ მნიშვნელობას: მათ საშუალო არითმეტიკულს და საშუალო გეომეტრიულს. შეეცადეთ, რომ ფუნქციამ მხოლოდ ერთხელ გადახედოს სიას. განსაზღვრეთ ფუნქციის ტიპი. მოიყვანეთ გამოძახების მაგალითი.   
     
   mean xs = helpermean xs 0 1 0 +5  
   helpermean (x:xs) sum mult lenght= helpermean xs (x+sum) (x\*mult) (lenght+1)  
   helpermean [] sum mult lenght =(sum/lenght, mult \*\*(1/lenght))  
     
   mean :: Floating t => [t] -> (t, t)  
     
   mean [1..10]
4. განსაზღვრეთ maRali rigis *any* da *takeWhile* funqciebi standartul prelude failSi mocemuli maTi aRwerebis gamouyeneblad.  
   myany f xs = or (map f xs) +5  
     
   mytakewhile f [] = []  
   mytakewhile f (x:xs)  
    | f x = x: mytakewhile f xs  
    |otherwise = []
5. უძრავი ქონების სააგენტოში იყიდება ბინები- Flat, ოთახები- Room და კერძო სახლები-House. ბინა ხასიათდება სართულით, ფართობით და სახლის სართულების რაოდენობით. ოთახი ხასიათდება ამის გარდა კიდევ ფართობით (დამატებით მთელი ბინის ფართობისა). კერძო სახლი ხასიათდება მხოლოდ ფართობით. ანუ განსაზღვრულია მონაცემთა ტიპი, რომელიც წარმოადგენს უძრავი ქონების ობიექტებზე ინფორმაციას. ანუ მოცემული გვაქვს ტიპი:

data NedvObject = Flat Int Int Int | Room Int Int Int Int | House Int deriving (Eq,Show)

-- Flat sarTuli, farTobi, sarTulianoba | Room sarTuli, farTobi, sarTulianob,a oTaxis farTobi | House ფართობი

მონაცემთა ბაზაში ინახება მნიშვნელობების წყვილები, რომელთაგან პირველი წარმოადგენს უძრავ ობიექტს, მეორე–მის ფასს.

განსაზღვრეთ შემდეგი ფუნქციები:

1. GetRoom - მონაცემთა ბაზიდან არჩევს ოთახებს;
2. getFlatByPrice - ბაზიდან ირჩევს ბინებს, რომელთა ფასი მოცემულის ტოლია;
3. getExceptBounds, irCevs binebs mocemul sarTulze;
4. getByType - მეორე არგუმენტად გადაეცემა სტრიქონი, ფასი და ბაზიდან შესაბამის ინფორმაციას იღებს. მაგალითად,

**\*Main>** getByType [(Flat 3 100 10,1000),(Room 4 120 9 20,1000),(House 200,1000),(Flat 1 100 10,900)] “Room” 1001

[(Room 4 120 9 20,1000)]

it :: [(NedvObject, Int)]

it :: [(NedvObject, Int)]

1) =17  
getRoom :: [(NedvObject,Int)] -> [(NedvObject,Int)]  
getRoom [] = []  
getRoom ((Room x y z i, price):xs) = (Room x y z i, price):getRoom xs  
getRoom (\_:xs) = getRoom xs  
  
2)  
getFlatByPrice :: [(NedvObject,Int)] -> Int -> [(NedvObject,Int)]  
getFlatByPrice [] \_ = []  
  
getFlatByPrice ((Flat a b c,y):xs) price =  
 if y == price then (Flat a b c,y):getFlatByPrice xs price  
 else getFlatByPrice xs price  
getFlatByPrice (\_:xs) price = getFlatByPrice xs price  
3)  
getExceptBounds :: [(NedvObject,Int)] -> Int -> [(NedvObject,Int)]  
getExceptBounds [] \_ = []  
getExceptBounds ((Flat floor area totalfloors , price ):xs) targetFloor =  
 if floor == targetFloor then (Flat floor area totalfloors, price): getExceptBounds xs targetFloor  
 else getExceptBounds xs targetFloor  
getExceptBounds (\_:xs) targetFloor = getExceptBounds xs targetFloor  
4)

getByType :: [(NedvObject,Int)] -> [Char]-> Int -> [(NedvObject,Int)]  
getByType [] \_ \_ = []  
getByType (x:xs) t targetPrice = case x of  
 (Flat xx y z, price) -> if t == "Flat" && price < targetPrice then x:getByType xs t targetPrice  
 else getByType xs t targetPrice  
 (Room xx y z zz, price) -> if t == "Room" && price < targetPrice then x:getByType xs t targetPrice else getByType xs t targetPrice  
 (House y , price) -> if t == "House" && price < targetPrice then x:getByType xs t targetPrice  
 else getByType xs t targetPrice