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

**ქულების განაწილება:** 1 - **5**ქ., 2 - **5** ქ., 3 - **5** ქ., 4 - **5** ქ., 5 - 1**5** ქ. =36 **ქულათა ჯამი:** **40** ქულა

**გამოცდის ხანგრძლივობა: 2 საათი.**

**ბილეთი**

1. რეკურსიის 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.

--map ფუნქცია პირველ პარამეტრად იღებს მოქმედებას(ფუნქციას) რომელსაც ახორციელებს სიის ყველა წევრზე და აბრუნებს ამ სიას +5

mymap :: (t -> a) -> [t] -> [a]

mymap \_ [] = []

mymap a (x:xs) = [(a x)] ++ mymap a xs

-- \*Main> mymap (3+) [1,2,3,4]

-- [4,5,6,7]

1. სიის კონსტრუქტორის გამოყენებით. დაწერეთ მაღალი რიგის საბიბლიოთეკო myfilter ფუნქცია, რომელიც ირჩევს სიიდან ყოველ ელემენტს, რომელიც პრედიკატს აკმაყოფილებს განსაზღვრეთ ფუნქციის ტიპი. მოიყვანეთ გამოძახების მაგალითი.

--filter ფუნქცია პირველ პარამეტრად იღებს მოქმედებას(if) რომელსაც მიხედვითაც იფილტრება სია +5

myfilter :: (a -> Bool) -> [a] -> [a]

myfilter \_ [] = []

myfilter a (x:xs) = if a x then [x] ++ myfilter a xs else [] ++ myfilter a xs

-- \*Main> myfilter (>2) [1,2,3,4]

-- [3,4]

1. განსაზღვრეთ რეკურსიულად და სიის კონსტრუქტორის გამოყენებით ფუნქცია, რომელიც შესასვლელზე ღებულობს სიას და აბრუნებს ორ მნიშვნელობას: თავდაპირველი სიიდან ლუწი რიცხვების სიას გაერთიანებულს კენტი რიცხვების სიასთან და დადებითი რიცხვების სიას გაერთიანებულს კენტი რიცხვების სიასთან თავდაპირველ სიაში მათი თანმიმდევრობის შენარჩუნებით. განსაზღვრეთ ფუნქციის ტიპი და მოიყვანეთ გამოძახების მაგალითი.

ff :: Integral a => [a] -> ([a], [a]) +5

ff a = (ff1 a ++ ff2 a , ff3 a)

ff1 [] = []

ff1 (x:xs) = if x`mod`2==0 then [x] ++ ff1 xs else [] ++ ff1 xs

ff2 [] = []

ff2 (x:xs) = if x`mod`2==1 then [x] ++ ff2 xs else [] ++ ff2 xs

ff3 s = [n | n<-s, n>0] ++ [n | n<-s, n`mod`2==1]

-- \*Main> ff [1,-2,3,4,-5,-8,51,-32,54,-21]

-- ([-2,4,-8,-32,54,1,3,-5,51,-21],[1,3,4,51,54,1,3,-5,51,-21])

1. gansazRvreT maRali rigis *all* da *dropWhile* funqciebi standartul prelude failSi mocemuli maTi aRwerebis gamouyeneblad.

myall :: (t -> Bool) -> [t] -> Bool +5

myall \_ [] = True

myall a (x:xs) = if a x then myall a xs else False

-- \*Main> myall (even) [2,8,50,74,92]

-- True

mydropWhile :: (t -> Bool) -> [t] -> [t]

mydropWhile \_ [] = []

mydropWhile a (x:xs) = if a x then mydropWhile a xs else [x] ++ [n | n<- xs]

-- \*Main> mydropWhile (<5) [1,2,3,4,5,1,72,-2]

-- [5,1,72,-2]

1. 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. -4

განსაზღვრეთ შემდეგი ფუნქციები:

1. getFlat monacemTa bazidan irCevs binebs;
2. getRoomByPrice - ბაზიდან ირჩევს ოთახებს, რომელთა ფასი მოცემულზე მეტია;
3. getExceptBounds, ირჩევს მონაცემთა ბაზიდან ბინებს, რომლებიც არ მდებარეობს პირველ და ბოლო სართულებზე.
4. getByType - მეორე არგუმენტად გადაეცემა სტრიქონი 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"

[(Room 4 120 9 20,1000)]

it :: [(NedvObject, Int)]

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

getFlat s = [n | n <- s, isFlat n]

isFlat (Flat \_ \_ \_) = True

isFlat (Room \_ \_ \_ \_) = False

isFlat (House \_) = False

--ფასი არააქვს ამიტომ ფართობზე გავაკეთებ ??????

getRoomByPrice far s = [n | n <- s, isRoomByPrice far n]

isRoomByPrice \_ (Flat \_ \_ \_) = False

isRoomByPrice far (Room \_ far1 \_ \_) = far < far1

isRoomByPrice \_ (House \_) = False

getExceptBounds s = [n | n <- s, isExceptBounds n]

isExceptBounds (Flat sarTuli \_ sarTulianoba) = (sarTuli /= 1) && ( sarTuli /= sarTulianoba)

isExceptBounds (Room \_ \_ \_ \_) = False

isExceptBounds (House \_) = False