ბილეთი======29

1. map :: (a -> b) -> [a] -> [b] +5

map f [] = []

map f (x:xs) = f x : map f xs

მაგ

map square [1 .. 7] = [1, 4, 9, 16, 25, 36, 49]

ასევე

const fastfoodloverssum= (sia) => { const results = []; for (let i = 0; i <sia.length; i++) { const person =sia[i];

if (fastfoodlovers(person.name)) { person.fastfoodlovers = true; } else { person.fastfoodlovers = false; } results.push(person); }

return results;};

2.

fun myfilter p nil = nil +5

| myfilter p (x::xs) =

if p x

then x :: myfilter p xs

else myfilter p xs

(\* val filter = fn : ('a -> bool) -> ('a list -> 'a list) \*)

…

მაგ

const people = [ { name: ‘nini’, age: 16 }, { name: ‘Mari’, age: 19 }, { name: ‘Nika’, age: 20 }, { name: ‘Giorgi’, age: 18 },];

const peopleAbove18 = (collection) => { return collection .filter((person) => person.age >= 18);}

3 +3

2

3. sumAndCount :: (Num a) => [a] -> (a, Int)

sumAndCount xs = foldr (\x (s, c) -> (s + x, c + 1)) (0, 0) xs

average :: (Num a) => [a] -> a

average [] = error ""

average xs = total / (fromIntegral counter) where

(total, counter) = sumAndCount xs

4. +5

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

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

takeWhileInc p xs = case zs of [] -> error "not found"

(z:\_) -> ys ++ [z]

where

(ys, zs) = span p xs

My\_any ::(a->bool)-> [a] ->bool

My\_any = undefined

5. =11

Flat- floor(int), area(int),allfloor(int)

Room-, whichfloor(int),area(int),allfloor(int), roomarea(int)

House- housearea(double)

//ბაზიდან სახლის არჩევა

getRoom :: [(NedvObject,Int)] -> [(Nedvobject,Int)]

getRoom [] = []

getRoom ((House x,y):xs) = (Room x,y):getRoom xs

getRoom (\_:xs) = getRoom xs

//getFlatByPrice - ბაზიდან ირჩევს ბინებს, რომელთა ფასი მოცემულის ტოლია

getByPrice :: [(NedvObject,Int)] -> Int -> [(NedvObject,Int)]

getByPrice [] \_ = []

getByPrice ((House a,y):xs) price =

if y=price then (House a,y):getByPrice xs price = =

else getByPrice xs price

getByPrice ((Flat a b c,y):xs) price =

if y=price then (Flat a b c,y):getByPrice xs price

else getByPrice xs price

getByPrice ((Room a b c d,y):xs) price =

if y=price then (Room a b c d,y):getByPrice xs price

else getByPrice xs price

// ირჩებს ბინებს მოცემულ სართულზე

getExceptBounds :: [(NedvObject,Int)] -> [(NedvObject,Int)]

getExceptBounds [] = []

getExceptBounds ((Flat x y z,a):xs) =

if (x/=z)&&(x/=1) then (Flat x y z,a):getExceptBounds xs

else getExceptBounds xs

getExceptBounds (\_:xs) = getExceptBounds xs