ბილეთი??????????????

გამოცდა, ილკინ გოჯაევი =21

1 ) განსაზღვრეთ map ფუნქცია ორნაირად (სიის კონსტრუქტორით და რეკურსიით):

myMap1 f xs = [f x | x <- xs] ტიპი, გამოძახება +3

myMap2 f [] = []

myMap2 f (x:xs) = f x : myMap2 f xs

2) განსაზღვრეთ filter ფუნქცია ორნაირად (სიის კონსტრუქტორით და რეკურსიით):

myFilter1 p xs = [x | x<-xs, p x] ტიპი, გამოძახება +3

myFilter2 p [] = []

myFilter2 p (x:xs) = if (p x) then x:myFilter2 p xs

else myFilter2 p xs

**5**) =15

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

-- Flat sarTuli farTobi sarTulianoba | Room sarTuli farTobi sarTulianoba

-- oTaxis farTobi | House farTobi

data Requirements = Type String | MinArea Int | MaxPrice Int | NeedFloor Int

-- getHouses monacemTa bazidan irCevs kerZo saxlebs

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

getHouses [] = []

getHouses ((House x,y):xs) = (House x,y):getHouses xs

getHouses (\_:xs) = getHouses xs

--getByPrice bazidan irCevs obieqtebs, romelTa fasi mocemulze naklebia

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

--getByLevel irCevs binebs mocemul sarTulze

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

getByLevel [] \_ = []

getByLevel ((Flat x y z,a):xs) floor =

if x==floor then (Flat x y z,a):getByLevel xs floor

else getByLevel xs floor

getByLevel (\_:xs) floor = getByLevel xs floor

--getExceptBounds irCevs binebs Sua sarTulebze

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

--query irCevs moTxovnaTa siis Sesabamis obieqtebs

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

getByArea [] \_ = []

getByArea ((Flat x y z,a):xs) area =

if y>=area then (Flat x y z,a):getByArea xs area

else getByArea xs area

getByArea ((Room x y z zz,a):xs) area =

if y>=area then (Room x y z zz,a):getByArea xs area

else getByArea xs area

getByArea ((House y,a):xs) area =

if y>=area then (House y,a):getByArea xs area

else getByArea xs area

getByType :: [(NedvObject,Int)] -> String -> [(NedvObject,Int)]

getByType [] \_ = []

getByType (x:xs) t = case x of

(Flat xx y z,a) -> if t=="Flat" then x:getByType xs t

else getByType xs t

(Room xx y z zz,a) -> if t=="Room" then x:getByType xs t

else getByType xs t

(House y,a) -> if t=="House" then x:getByType xs t

else getByType xs t

--

query :: [(NedvObject,Int)] -> [Requirements] -> [(NedvObject,Int)]

query [] \_ = []

query x [] = x

query x (y:ys) = case y of

MaxPrice price -> query (getByPrice x price) ys

NeedFloor floor -> query (getByLevel x floor) ys

MinArea area -> query (getByArea x area) ys

Type t -> query (getByType x t) ys

--getHouses [(Flat 3 100 10,1000),(Room 4 120 9 20,500),(House 200,2000),(Flat 1 100 10,900)]

--getByPrice [(Flat 3 100 10,1000),(Room 4 120 9 20,500),(House 200,2000),(Flat 1 100 10,900)] 1001

--getByLevel [(Flat 3 100 10,1000),(Room 4 120 9 20,500),(House 200,2000),(Flat 1 100 10,900)] 1

--getExceptBounds [(Flat 3 150 10,1000),(Room 4 120 9 20,500),(House 200,2000),(Flat 1 100 10,900)]

--query [(Flat 3 110 10,1000),(Room 4 120 9 20,500),(House 200,2000),(Flat 1 100 10,900)] [(MinArea 101),(Type "Flat"),(NeedFloor 3)]