მაგალითი

უძრავი ქონების სააგენტოში იყიდება ბინები, ოთახები და კერძო სახლები. ბინა ხასიათდება სართულით, ფართობით და სახლის სართულების რაოდენობით. ოთახი ხასიათდება ამის გარდა კიდევ ფართობით (დამატებით მთელი ბინის ფართობისა). კერძო სახლი ხასიათდება მხოლოდ ფართობით.

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

განსაზღვრეთ მონაცემთა ტიპი, რომელიც წარმოადგენს უძრავი ქონების ობიექტებზე ინფორმაციას. განსაზღვრეთ შემდეგი ფუნქციები:

1) getHouses, არჩევს მონაცემთა ბაზიდან მხოლოდ კერძო სახლებს.

2) getByPrice, არჩევს მონაცემთა ბაზიდან მხოლოდ უძრავი ქონების იმ ობიექტებს, რომელთა ფასი ნაკლებია მითითებულზე.

3) getByLevel, ირჩევს მონაცემთა ბაზიდან ბინებს, რომლებიც მოცემულ სართულზე მდებარეობს.

4) getExceptBounds, ირჩევს მონაცემთა ბაზიდან ბინებს, რომლებიც არ მდებარეობს პირველ და ბოლო სართულებზე.

შეიმუშავეთ მონაცემების ტიპი, რომელიც წარმოადგენს უძრავი ქონების ობიექტებზე სხვადასხვა მოთხოვნებს: უძრავი ქონების სასურველი ტიპის ობიექტს, მინიმალურ ფართობს, მაქსიმალურ ფასს, შეზღუდვას სართულზე. შექმენით ფუნქცია query, რომელიც მოთხოვნების სიის მიხედვით მონაცემთა ბაზიდან ამოიღებს უძრავი ქონების იმ ობიექტებს, რომლებიც აკმაყოფილებს ყველა მოთხოვნას.

ამოხსნა:

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

-- getFlat monacemTa bazidan irCevs binebs;

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

getFlat [] = []

getFlat ((Flat x a b,y):xs) = (Flat x a b,y): getFlat xs

getFlat (\_:xs) = getFlat xs

-- getRoom monacemTa bazidan irCevs ოთახებს

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

getRoom [] = []

getRoom ((Room x a b c,y):xs) = (Room x a b c,y): getRoom xs

getRoom (\_:xs) = getRoom 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

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

getFlatByPrice [] \_ = []

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

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

else getFlatByPrice xs price

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

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

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

[(House 200,1000)]

it :: [(NedvObject, Int)]

--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

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

[(Flat 3 100 10,1000),(Flat 1 100 10,900)]

it :: [(NedvObject, Int)]

**\*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)]

--

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

ან

ssia :: [(NedvObject, Int)]

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

>getHouses ssia

>getFlat ssia

>getRoom ssia

--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)]

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

getByType [] \_ \_ = []

getByType (x:xs) t price= case x of

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

else getByType xs t price

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

else getByType xs t price

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

else getByType xs t price

**\*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)]

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

[(Flat 3 100 10,1000),(Flat 1 100 10,900)]

it :: [(NedvObject, Int)]