Contracts

Contracts and semi-strict boolean operators

Imagine the following (somewhat abstract and simplified) model of a person’s financial situation:

In this world, there are people (i.e. objects of type PERSON) who are defined by their name and their age. Each person of age at least 18 is allowed to open a bank account (i.e. an object of type BANK_ACCOUNT), which stays the same for the rest of the person’s life.

Such an account has a certain balance, which is not allowed to fall below zero (i.e. no person is allowed to have debts).

A person can deposit money to and withdraw money from an account if and only if he/she knows the correct PIN (there is no credit or bank card needed).

A PIN is an integer with at least 4 and a maximum of 6 digits (for simplicity, we do not allow leading zeros). When a new bank account is opened, the PIN is automatically set to "123456" but can be changed by any person knowing the current PIN.

A bank account can either be open or closed. By default, it is open (active). If it is not active, no actions can be performed on the account, i.e. no money can be deposited or withdrawn and the PIN cannot be changed until the account is re-opened.

Furthermore, a person living in this world also has to pay taxes, of course. The amount payable depends solely on the balance of the person’s bank account, namely it is one hundredth of the current balance. If a person does not have an account or the account is closed, no taxes are due.

The above situation is represented by the classes BANK_ACCOUNT and PERSON, which are attached below.

Read through the classes carefully and make sure you understand what each feature does.

Task 1 - Find contracts

For each of the features in the two classes, try to find contracts so that the model corresponds to the above description.

As an example, the preconditions and postconditions of feature make of class PERSON are given.

The number of dotted lines does not necessarily reflect the number of contracts.

Task 2 - strict vs semi-strict

- What kind of semi-strict boolean operators are there?
- What is the difference between strict and semi-strict boolean operators?
- Give a couple of examples of a situation where one should use semi-strict operators!
class PERSON

create

make

feature -- creation

make(a_first_name, a_last_name: STRING; a_age: INTEGER)
-- create a new person with given age and name

require
  valid_first_name: a_first_name /= Void and then not a_first_name.is_empty
  valid_last_name: a_last_name /= Void and then not a_last_name.is_empty
  valid_age: a_age >= 0

do
  set_first_name(a_first_name)
  set_last_name(a_last_name)
  set_age(a_age)

ensure
  first_name_set: first_name.is_equal(a_first_name)
  last_name_set: last_name.is_equal(a_last_name)
  age_set: age = a_age

end

feature

open_account(a_initial_amount: INTEGER)
-- opens an account for the person

require

local
  b: BANK_ACCOUNT

do
  create b.make (Current, a_initial_amount)
  account := b

ensure

have_birthday
-- increases age by 1

require
do
  age := age + 1
ensure
  ...........................................................
  ...........................................................
end

has_account: BOOLEAN
  -- Does the person have a bank account?
require
  ...........................................................
  ...........................................................
do
  Result := account /= Void
ensure
  ...........................................................
  ...........................................................
end

taxes: INTEGER
  -- How many taxes does this person have to pay?
require
  ...........................................................
  ...........................................................
do
  if
    has_account and then account.active
  then
    Result := account.balance // 100
  else
    Result := 0
  end
ensure
  ...........................................................
  ...........................................................
end

feature -- setting
set_age(a_age: INTEGER)
require
  ...........................................................
  ...........................................................
do
  age := a_age
ensure
set_first_name(a_name: STRING)
require
do
        first_name := a_name.twin
ensure
end
set_last_name(a_name: STRING)
require
do
        last_name := a_name.twin
ensure
end

feature -- attributes

    age: INTEGER
        -- Age of the person

    first_name, last_name: STRING
        -- first and last name of the person

    account: BANK_ACCOUNT
        -- bank account of the person

invariant
end
class 
    BANK_ACCOUNT

create
    make

feature -- creation

make(a_owner: PERSON; a_initial_balance: INTEGER)
    -- crate a new account with a specific owner and balance
    require
        ..............................................................
        ..............................................................
    do
        owner := a_owner
        balance := a_initial_balance
        active := True
        pin := 123456
    ensure
        ..............................................................
        ..............................................................
        ..............................................................
        ..............................................................
    end

feature -- modification

withdraw(a_sum, a_pin: INTEGER)
    -- withdraw 'a_sum' from the account
    require
        ..............................................................
        ..............................................................
    do
        balance := balance - a_sum
    ensure
        ..............................................................
        ..............................................................
    end

deposit(a_sum, a_pin: INTEGER)
    -- deposit 'a_sum' on to the account
    require
        ..............................................................
        ..............................................................
        ..............................................................
        ..............................................................

do
    balance := balance + a_sum
ensure
    .................................................................
    .................................................................
end

change_pin(a_old_pin, a_new_pin: INTEGER)
    -- change the pin
require
    .................................................................
    .................................................................
    .................................................................
do
    pin := a_new_pin
ensure
    .................................................................
    .................................................................
end

close
    -- close the account
require
    .................................................................
    .................................................................
do
    active := False
ensure
    .................................................................
    .................................................................
end

open
    -- open the account
require
    .................................................................
    .................................................................
do
    active := True
ensure
    .................................................................
    .................................................................
end
feature  --  attributes

  balance: INTEGER
     --  balance of the account

  pin: INTEGER
     --  pin of the account

  active: BOOLEAN
     --  state of the account

  owner: PERSON
     --  owner of the account

invariant

end