Systems Development: Object Oriented Analysis and Design

(H172 35)

Introducing Inheritance

Employees Walkthrough

Step 2 – creating an inheritance hierarchy

Much of the code in BasePlusCommissionEmployee is similar, if not identical, to the code in CommissionEmployee, therefore we are now going to use inheritance and declare that BasePlusCommissionEmployee extends class CommissionEmployee.

We are going to implement the following scenario:

CommissionEmployee

- firstName:string

- lastName:string

- socialSecurityNumber:string

- grossSales:decimal

- commissionRate:decimal

+ <<property>> FirstName:string

+ <<property>> LastName:string

+ <<property>> SocialSecurityNumber:string

+ <<property>> GrossSales:decimal

+ <<property>> CommissionRate:string

+ CommissionEmployee(first:string, last:string, ssn:string, sales:decimal, rate:decimal)

+ Earnings():decimal

+ ToString():string

BasePlusCommissionEmployee

- baseSalary:decimal

+ <<property>> BaseSalary:decimal

+ BasePlusCommissionEmployee(first:string, last:string, ssn:string, sales:decimal, rate:decimal, salary:decimal)

+ Earnings():decimal

+ ToString():string

BasePlusCommissionEmployee acquires all the capabilities of CommissionEmployee and additionally has instance variable baseSalary. Therefore we are going to implement BasePlusCommissionEmployee as inheriting from CommissionEmployee.

1. Update the BasePlusCommissionEmployee class to inherit from class CommissionEmployee

Hint: use the colon(:) notation in the class declaration to indicate inheritance

1. Make the following updates to the BasePlusCommissionEmployee class methods

BasePlusCommissionEmployee Constructor

The constructor of CommissionEmployee is not inherited, therefore make sure the BasePlusCommissionEmployee calls the base class CommissionEmployee constructor using a constructor initialiser.

Hint: :base(first, last, ssn, sales, rate)

(Each derived class constructor *must* implicitly or explicitly call its base-class constructor to ensure that the instance variables inherited from the base class are initialized properly. Implement BasePlusCommissionEmployee’s six-parameter constructor to explicitly call class CommissionEmployee’s five-parameter constructor to initialize the CommissionEmployee portion of a BasePlusCommissionEmployee object.)

Method Earnings()

BasePlusCommissionEmployee has it’s own implementation of method Earnings() as defined in Step 1, therefore we need to *override* this method.

commisionRate and grossSales will be underlined red stating that they are not accessible – see “Issues” below

Method ToString()

BasePlusCommissionEmployee has it’s own implementation of method ToString() as defined in Step 1, therefore we need to *override* this method.

Variables will be underlined in red stating that they are not accessible – see “Issues” below

**Issues**

Error - the Earnings() method and ToString() method in BasePlusCommissionEmployee refer to **private** instance variables in the base class CommissionEmployee. *private* members are inherited by the derived class but are not accessible in the derived class.

Error - We cannot override the base class’s Earnings() method because it was not explicitly marked as “virtual”. The override modifier declares that a derived class method overrides a virtual or abstract base class method. (This modifier also implicitly declares the derived class method virtual and allows it to be overridden in the derived classes further down the inheritance hierarchy).