CSULB
CECS277

Project6

PartI

(Payroll System Modification)

Modify the payroll system of lab5 to include an additional Employee subclass **PieceWorker** that represents an employee whose pay is based on the number of pieces of merchandise produced. Class **PieceWorker** should contain private instance variables wage (to store the employee's wage per piece) and pieces (to store the number of pieces produced). Provide a concrete implementation of method earnings in class **PieceWorke**r that calculates the employee's earnings by multiplying the number of pieces produced by the wage per piece.

Create an array of **Employee** variables to store references to objects of each concrete class in the new **Employee** hierarchy. For each **Employee**, display its String representation and earnings.

Run your code, save your code and keep a sample of the output

PartI

(Accounts Payable System Modification)

Now modify the accounts payable application of lab6 to include the complete functionality of the payroll application. The application should still process two **Invoice** objects, but now should process one object of each of the four **Employee** subclasses. If the object currently being processed is a **Base- PlusCommissionEmployee**, the application should increase the BasePlusCommissionEmployee's base salary by 10%. Finally, the application should output the payment amount for each object.

Complete the following steps to create the new application:

a. Modify classes **HourlyEmployee** and **CommissionEmployee** to place them in the **Payable** hierarchy as subclasses of the version of **Employee** that implements **Payable**. [*Hint:* Change the name of method **earnings** to **getPaymentAmount** in each subclass so that the class satisfies its inherited contract with interface **Payable**.]
b. Modify class **BasePlusCommissionEmployee** such that it extends the version of class **CommissionEmployee** created in part (a).
c. Modify **PayableInterfaceTest** to polymorphically process two **Invoice**S, one **SalariedEmployee**, one **HourlyEmployee**, one **CommissionEmployee** and one **Base-PlusCommissionEmployee**. First output a String representation of each Payable object. Next, if an object is a **BasePlusCommissionEmployee**, increase its base salary by 10%.

Finally, output the payment amount for each **Payable object**.

For both parts, override the methods defined in Object class that we covered during lectures (toString, equals, hashCode, compareTo)

Properly document your codes. Draw the UML (s). Give a sample output.

Save everything in a word document and upload.