Name of the Course : Java for beginners: Step–by–step hands-on guide to Java

Level : Difficult

Tool Stack : Abstarction and Encapsulation, Polymorphism, Inheritance and access specifiers

Problem Statement :

Global Engineering is one of the fastest growing company. It needs to automate the transactions performed in the organization. As start up, they need to automate the Employee management system.

Description : You are provided with a public class Employee with protected attributes :

int employeeId

String employeeName

double salary

Appropriate public getters and setters are already written.

Write a public 2 argument constructor with arguments – employeeId,and employeeName.

Write a public abstract method calculateSalary() in Employee class as,

public abstract void calculateSalary()

You are provided with a public class PermanentEmployee with private attribute :

double basicPay

Appropriate public getters and setters are already written.

Make this class PermanentEmployee to inherit the Employee class.

Write a public 3 argument constructor with arguments – employeeId, employeeName and basicPay.

Implement the calculateSalary method in Employee class as

salary = basicPay – PF amount

Here PF Amount = basicPay \* 0.12; Set this value to the salary attribute.

You are provided with a public class TemporaryEmployee with private attribute :

int hoursWorked

int hourlyWages

Appropriate public getters and setters are already written.

This class TemporaryEmployee should inherit the Employee class.

Write a public 4 argument constructor with arguments – employeeId, employeeName, hoursWorked and hourlyWages.

Implement the calculateSalary method in Employee class as

salary = hoursWorked \* hourlyWages

Set this value to the salary attribute.

You are provided with a public class Loan

A method calculateLoanAmount is provided as shown below :

public double calculateLoanAmount(Employee employeeObj)

This method should calculate the loan amount and return that amount. Provide the implementation for this method as mentioned below

Loan amount is calculated as follows :

If the Employee object is of type PermanentEmployee the loan amount should be 15% of the salary.

If the Employee object is of type TemporaryEmployee the loan amount should be 10% of the salary.

You are provided with a public class Main which has the main method.

Check the correctness of the methods written in these classes.

Note : All class, methods needs to be declared as public

**Code:**

**package** com.LoanEligibility;

**public** **abstract** **class** Employee {

**private** **int** employeeId;

**private** String employeeName;

**private** **double** salary;

**public** **int** getEmployeeId() {

**return** employeeId;

}

**public** **void** setEmployeeId(**int** employeeId) {

**this**.employeeId = employeeId;

}

**public** String getEmployeeName() {

**return** employeeName;

}

**public** **void** setEmployeeName(String employeeName) {

**this**.employeeName = employeeName;

}

**public** **double** getSalary() {

**return** salary;

}

**public** **void** setSalary(**double** salary) {

**this**.salary = salary;

}

**public** Employee(**int** employeeId,String employeeName)

{

**this**.employeeId=employeeId;

**this**.employeeName=employeeName;

}

**public** **abstract** **void** calculateSalary();

}

**package** com.LoanEligibility;

**public** **class** PermanentEmployee **extends** Employee {

**private** **double** basicPay;

**public** PermanentEmployee(**int** employeeId, String employeeName,**double** basicPay) {

**super**(employeeId, employeeName);

**this**.basicPay=basicPay;

}

**public** **double** getBasicPay() {

**return** basicPay;

}

**public** **void** setBasicPay(**double** basicPay) {

**this**.basicPay = basicPay;

}

@Override

**public** **void** calculateSalary() {

**double** PF\_Amount = basicPay \* 0.12;

**double** salary = basicPay-PF\_Amount;

**this**.setSalary(salary);

}

}

**package** com.LoanEligibility;

**public** **class** TemporaryEmployee **extends** Employee {

**private** **int** hoursWorked;

**private** **int** hourlyWages;

**public** TemporaryEmployee(**int** employeeId, String employeeName,**int** hoursWorked,**int** hourlyWages) {

**super**(employeeId, employeeName);

**this**.hourlyWages=hourlyWages;

**this**.hoursWorked=hoursWorked;

}

**public** **int** getHoursWorked() {

**return** hoursWorked;

}

**public** **void** setHoursWorked(**int** hoursWorked) {

**this**.hoursWorked = hoursWorked;

}

**public** **int** getHourlyWages() {

**return** hourlyWages;

}

**public** **void** setHourlyWages(**int** hourlyWages) {

**this**.hourlyWages = hourlyWages;

}

@Override

**public** **void** calculateSalary() {

**double** salary = hoursWorked \* hourlyWages;

**this**.setSalary(salary);

}

}

**package** com.LoanEligibility;

**public** **class** Loan {

**public** **double** calculateLoanAmount(Employee employeeObj) {

**double** loanAmount=0;

**if**(employeeObj **instanceof** PermanentEmployee)

{

employeeObj.calculateSalary();

loanAmount=0.15\*employeeObj.getSalary();

}

**if**(employeeObj **instanceof** TemporaryEmployee)

{

employeeObj.calculateSalary();

loanAmount=0.10\*employeeObj.getSalary();

}

**return** loanAmount;

}

}

**package** com.LoanEligibility;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

Loan loan=**new** Loan();

Employee pe=**new** PermanentEmployee(1, "Ritika", 50000);

Employee te=**new** TemporaryEmployee(56, "Sohan", 200, 100);

**double** loanAmount\_pe=loan.calculateLoanAmount(pe);

**double** loanAmount\_te=loan.calculateLoanAmount(te);

System.***out***.println(pe.getEmployeeName()+" "+"loan amount is "+loanAmount\_pe);

System.***out***.println(te.getEmployeeName()+" "+"loan amount is "+loanAmount\_te);

}

}

Junit Testing

**package** com.LoanEligibility.test;

**import** **static** org.junit.jupiter.api.Assertions.\*;

**import** org.junit.Before;

**import** org.junit.jupiter.api.Test;

**import** com.LoanEligibility.Employee;

**import** com.LoanEligibility.Loan;

**import** com.LoanEligibility.PermanentEmployee;

**import** com.LoanEligibility.TemporaryEmployee;

**class** MainTest {

@Test

**void** testCalculateLoanAmount() {

Loan loan=**new** Loan();

Employee p=**new** PermanentEmployee(1, "Ritika", 50000);

Employee t=**new** TemporaryEmployee(56, "Sohan", 200, 100);

p.calculateSalary();

t.calculateSalary();

**double** loanAmountP=loan.calculateLoanAmount(p);

*assertEquals*(6600.0, loanAmountP);

**double** loanAmountT=loan.calculateLoanAmount(t);

*assertEquals*(2000.0, loanAmountT);

}

@Test

**void** testCalculateSalary() {

Employee p=**new** PermanentEmployee(1, "Ritika", 50000);

Employee t=**new** TemporaryEmployee(56, "Sohan", 200, 100);

p.calculateSalary();

t.calculateSalary();

*assertEquals*(44000.0,p.getSalary());

*assertEquals*(20000.0,t.getSalary());

}

}

Test Data1

**Output:**

Ritika loan amount is 6600.0

Sohan loan amount is 2000.0

Learning outcome: Participant could able to learn how to use abstract class and abstract method as well as polymorphism, inheritance, encapsulation and access specifires.