Name of the Course : Complete Java SE8 Developer Bootcamp

Level : Difficult

Tool Stack : Java8 and Junit4

Problem Statement : Provide a code solution to accept employee details. Based on input, system will understand what kind of Employee i.e. Permanent or Contractual and arrange them accordingly using Inheritance , abstract class and collection concept.

Description : ***PQR & Company Ltd*** two kinds of employee, Permanent and Contractual .The Permanent Employee has monthly salary and PF which is 15% of salary. Contractual employee has contract period and contract amount. Every employee has id (for permanent ids are like P1001,P1002... and for contractual ids are like C1001,C1002... etc.), name, dept-name. The input for any employee is in comma(,) separate format eg:

1. Eric Miller,Finance,35000.00 -> for Permanent employee.
2. Roger Steven,Sales,5,750000.00 -> Contractual employee.

Id will be auto generated as shown above. Finally records will be stored in respective list. All employees need to pay tax which is 10% of annual salary. Finally you need to display all permanent employees in ascending order of employee id and contractual employees in descending order of id. You need create

1. class Employee with private member data

**String employeeId,**

**String employeeName,**

**String department**

Create getter/setter methods and constructors.

calculateTax()-> without body

override toString() in String.format("%-10s %-20s %-15s").

The class must implement Comparable interface

1. class PermanentEmployee inherited from Employee with private member data

**Double monthlySalary,**

**Double pf, (15% of monthlySalary)**

**Double tax,**

**static int idGen,**

Create getter/setter methods and constructors.

override calculateTax()

override toString() in String.format("%-15s %-15s"...)

static method generateId() : to generate id for permanent employees

1. class ContractEmployee inherited from Employee with private member data

**Integer contractPeriod,**

**Double contarctAmount,**

**Double tax,**

**static int idGenerator,**

Create getter/setter methods and constructors.

override calculateTax()

override toString() in String.format("%-15s %-15s"...)

static method generateId() : to generate id for permanent employees

1. class Main with method public static void main(String [] arg)-> In this method
2. Only one Employee reference variable is needed to declare.
3. Two lists (one for PermanentEmployee another for ContractEmployee) of type Employee needed to declare.
4. Asks to enter number of employees record to accept and then all employees in comma separate format.

Code:

**public** **abstract** **class** Employee **implements** Comparable<Employee>

{

**private** String employeeId;

**private** String employeeName;

**private** String department;

**public** Employee(String employeeId, String employeeName, String department) {

**super**();

**this**.employeeId = employeeId;

**this**.employeeName = employeeName;

**this**.department = department;

}

**public** Employee() {

**super**();

}

**public** String getEmployeeId() {

**return** employeeId;

}

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

**this**.employeeId = employeeId;

}

**public** String getEmployeeName() {

**return** employeeName;

}

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

**this**.employeeName = employeeName;

}

**public** String getDepartment() {

**return** department;

}

**public** **void** setDepartment(String department) {

**this**.department = department;

}

@Override

**public** String toString() {

String output=String.*format*("%-10s %-20s %-15s",employeeId,employeeName,department);

**return** output;

}

**public** **int** compareTo(Employee emp)

{

**return** **this**.employeeId.compareTo(emp.employeeId);

}

**public** **abstract** **void** calculateTax();

}

**import** java.text.DecimalFormat;

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

**private** Double monthlySalary;

**private** Double pfContribution;

**private** Double tax;

**private** **static** **int** *idgen*=1000;

**public** **static** String generateId()

{

String id="P";

*idgen*++;

id=id+*idgen*;

**return** id;

}

**public** PermanentEmployee() {

**super**();

// **TODO** Auto-generated constructor stub

}

**public** PermanentEmployee(String employeeId, String employeeName, String department,

Double monthlySalary, Double pfContribution) {

**super**(employeeId, employeeName, department);

**this**.monthlySalary = monthlySalary;

**this**.pfContribution = pfContribution;

}

**public** Double getMonthlySalary() {

**return** monthlySalary;

}

**public** **void** setMonthlySalary(Double monthlySalary) {

**this**.monthlySalary = monthlySalary;

}

**public** Double getPfContribution() {

**return** pfContribution;

}

**public** **void** setPfContribution(Double pfContribution) {

**this**.pfContribution = pfContribution;

}

**public** Double getTax() {

**return** tax;

}

**public** **void** setTax(Double tax) {

**this**.tax = tax;

}

**public** **void** calculateTax()

{

**double** annualSalary=monthlySalary\*12;

tax=annualSalary/10;

}

@Override

**public** String toString() {

String str=**super**.toString();

DecimalFormat df=**new** DecimalFormat("0.00");

String salary=df.format(monthlySalary);

String pf=df.format(pfContribution);

String tx=df.format(tax);

String output=str+String.*format*("%-10s %-10s %-10s",salary,pf,tx);

**return** output;

}

}

**import** java.text.DecimalFormat;

**public** **class** ContractEmployee **extends** Employee {

**private** Integer contractPeriod;

**private** Double contractAmount;

**private** Double tax;

**private** **static** **int** *idgen*=1000;

**public** **static** String generateId()

{

String id="C";

*idgen*++;

id=id+*idgen*;

**return** id;

}

**public** ContractEmployee() {

**super**();

// **TODO** Auto-generated constructor stub

}

**public** ContractEmployee(String employeeId, String employeeName, String department,Integer contractPeriod, Double contractAmount) {

**super**(employeeId, employeeName,department);

**this**.contractPeriod = contractPeriod;

**this**.contractAmount = contractAmount;

}

**public** Integer getContractPeriod() {

**return** contractPeriod;

}

**public** **void** setContractPeriod(Integer contractPeriod) {

**this**.contractPeriod = contractPeriod;

}

**public** Double getContractAmount() {

**return** contractAmount;

}

**public** **void** setContractAmount(Double contractAmount) {

**this**.contractAmount = contractAmount;

}

@Override

**public** String toString() {

DecimalFormat df=**new** DecimalFormat("0.00");

String amount=df.format(contractAmount);

String tx=df.format(tax);

String str=**super**.toString();

String output=str+String.*format*("%-10s %-10s %-10s ",contractPeriod,amount,tx);

**return** output;

}

**public** Double getTax() {

**return** tax;

}

**public** **void** setTax(Double tax) {

**this**.tax = tax;

}

@Override

**public** **void** calculateTax() {

**double** annualIncome=contractAmount/contractPeriod;

tax=annualIncome/10;

}

}

**import** java.util.ArrayList;

**import** java.util.Collections;

**import** java.util.List;

**import** java.util.Scanner;

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

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

Scanner scanner=**new** Scanner(System.***in***);

System.***out***.println("Enter Number of Employees.");

**int** empNumbers=Integer.*parseInt*(scanner.nextLine());

Employee emp=**null**;

List<Employee> parmanentList=**new** ArrayList<Employee>();

List<Employee> contractList=**new** ArrayList<Employee>();

System.***out***.println("Enter all Employees details...");

**for**(**int** i=0;i<empNumbers;i++)

{

String empInfo=scanner.nextLine();

String arr[]=empInfo.split(",");

**if**(arr.length==3)

{

**double** salary=Double.*parseDouble*(arr[2]);

**double** pf=salary\*0.15;

String id=PermanentEmployee.*generateId*();

emp=**new** PermanentEmployee(id,arr[0],arr[1],salary,pf);

emp.calculateTax();

parmanentList.add(emp);

}

**else** **if**(arr.length==4)

{

**int** period=Integer.*parseInt*(arr[2]);

**double** amount=Double.*parseDouble*(arr[3]);

String id=ContractEmployee.*generateId*();

emp=**new** ContractEmployee(id,arr[0],arr[1],period,amount);

emp.calculateTax();

contractList.add(emp);

}

}

Collections.*sort*(parmanentList);

Collections.*reverse*(contractList);

System.***out***.println("\nParmanent Employee List");

String output=String.*format*("%-10s %-20s %-15s %-10s %-10s %-10s","Id","Name","Department","Salary","PF","Tax");

System.***out***.println(output);

**for**(Employee employee:parmanentList)

System.***out***.println(employee);

System.***out***.println("\nContract Employee List");

output=String.*format*("%-10s %-20s %-15s %-10s %-10s %-10s","Id","Name","Department","Period","Amount","Tax");

System.***out***.println(output);

**for**(Employee employee:contractList)

System.***out***.println(employee);

}

}

Junit Testing

Test Data1

Enter Number of Employees.

9

Enter all Employees details...

Becky Robins,Admin,27500.00

Starla Jones,Sales,32500.00

Lillian Williams,Store,3,750000.00

Eric Berger,Sales,25000.00

Mike Rodney,Production,5,1000000.00

Emily Parker,Finance,3,600000.00

Tom Hilman,Sales,35000.00

Harry Noel,Purchase,5,1500000.00

Mergery Dugal,Admin,35000.00

Parmanent Employee List

Id Name Department Salary PF Tax

P1001 Becky Robins Admin 27500.00 4125.00 33000.00

P1002 Starla Jones Sales 32500.00 4875.00 39000.00

P1003 Eric Berger Sales 25000.00 3750.00 30000.00

P1004 Tom Hilman Sales 35000.00 5250.00 42000.00

P1005 Mergery Dugal Admin 35000.00 5250.00 42000.00

Contract Employee List

Id Name Department Period Amount Tax

C1004 Harry Noel Purchase 5 1500000.00 30000.00

C1003 Emily Parker Finance 3 600000.00 20000.00

C1002 Mike Rodney Production 5 1000000.00 20000.00

C1001 Lillian Williams Store 3 750000.00 25000.00

Learning outcome: Participant could able to use inheritance feature, abstract class, abstract method, collection in inheritance.