PART 01 : INVOICE

Create a class called Invoice that a hardware store might use to represent an invoice for an item sold at the store. An Invoice should include four pieces of information as instance variables. i. number(type String)

ii. description (type String)

iii. quantity of the item being purchased (type int)

iv. price per item (double).

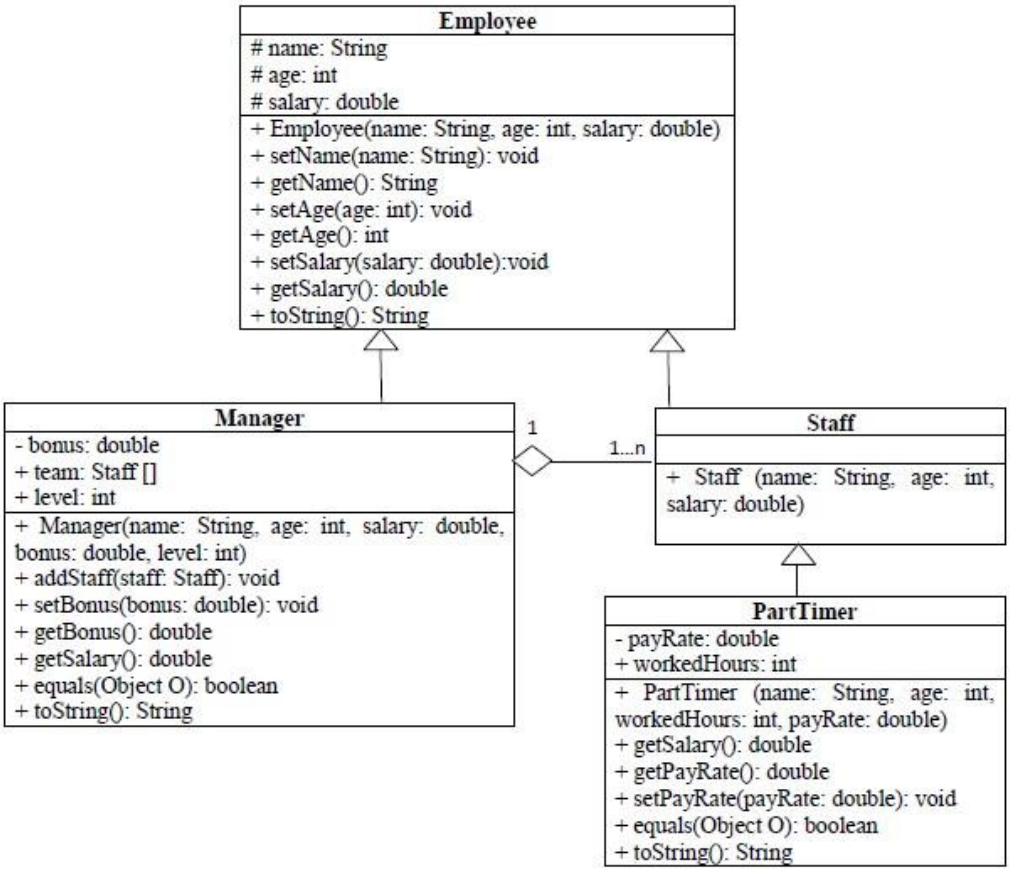
Your class should have a constructor that initializes the four instance variables. Provide a set and a get method for each instance variable. In addition, provide a method named getInvoiceAmount that calculates the invoice amount (i.e., multiplies the quantity by the price per item), then returns the amount as a double value. If the quantity is not positive, it should be set to 0. If the price per item is not positive, it should be set to 0.0. Write a test application named InvoiceTest that demonstrates class Invoice's capabilities.

PART 02 : INVOICE

Since classes are the building block of objects, class diagrams are the building blocks of UML. The various components in a class diagram can represent the classes that will actually be programmed, the main objects, or the interactions between classes and objects.

Implement the following classes hierarchy represented using the UML class diagram shown below. Consider the following details when implementing the classes:

- In the Manager class:
  - The team array should be initialized with 0 elements.
  - The toString method should be overridden such that it prints the details of the manager and the names of the staff members he manages.
  - The getSalary method is overridden such that it returns the salary plus the bonus.



  - The equals method is overridden such that it returns true if the two managers have the same level.

- In the PartTimer class:

- The getSalary method is overridden such that it returns the salary as the payRate multiplied by the workedHours.
- The equals method is overridden such that it returns true if the two part-timers have the same payRate.

- In your main class:
  - Create a method called printEmployeeDetails with the following header: public static void printEmployeeDetails (Employee [] e)
  - The method should ask the user to enter the employee name using an input terminal, then find the employee with the entered name in the array of employees and print his/her details by invoking the toString method.
- In your main method:
  - Create an array of 6 employees.
  - For the actual type of each employee, you should ask the user to choose the type of employee to using terminal.

PART 03 : BANK

You are required to design a banking system which consists of several branches in different locations and each branch has a number of registered accounts.

a. The system consists of branches with an ID and location.

b. Each branch has a number of accounts each representing an account for a client.

c. Each account has an ID, client name, and a balance which represents the amount of money in the account.

d. All accounts have an annual interest rate which represents the rate used to compute the annual interest for that account. Note that all accounts should share the same value of the annual interest rate (default 4.5%).

e. All data fields in the designed classes should be encapsulated.

f. You should provide the following features via the system

i. Ability to enter account info from the user each time a new account is created.

1. Make sure that the id entered by the user includes exactly six digits, if not it should keep on asking the user to enter a valid id, and

2. Make sure that the entered balance is not a negative value, if so it should keep on asking the user to enter a valid positive or zero balance.

ii. Ability to enter branch info from the user each time a new branch is created.

iii. Ability to compute the monthly interest rate.

(Monthly interest rate = annual interest rate / 12)

iv. Ability to compute monthly interest for the account.

(Monthly interest = balance * monthly interest rate).

v. Ability to withdraw a specified amount of money from the account. Note that the method should check whether the available balance is sufficient to perform the withdrawal and print the following message if it fails: “Withdrawal failed: no sufficient balance”.

vi. Ability to deposit a specified amount of money to the account.

vii. Ability to obtain info of the number of accounts in each branch, and the total number of accounts in all the branches.

viii. Ability to print the details of any account in any of the branches.

ix. Ability to print the details of all branches in the system.

- Draw the class diagram of for the given question.
- Write a Java program that implements your classes with given requirements.