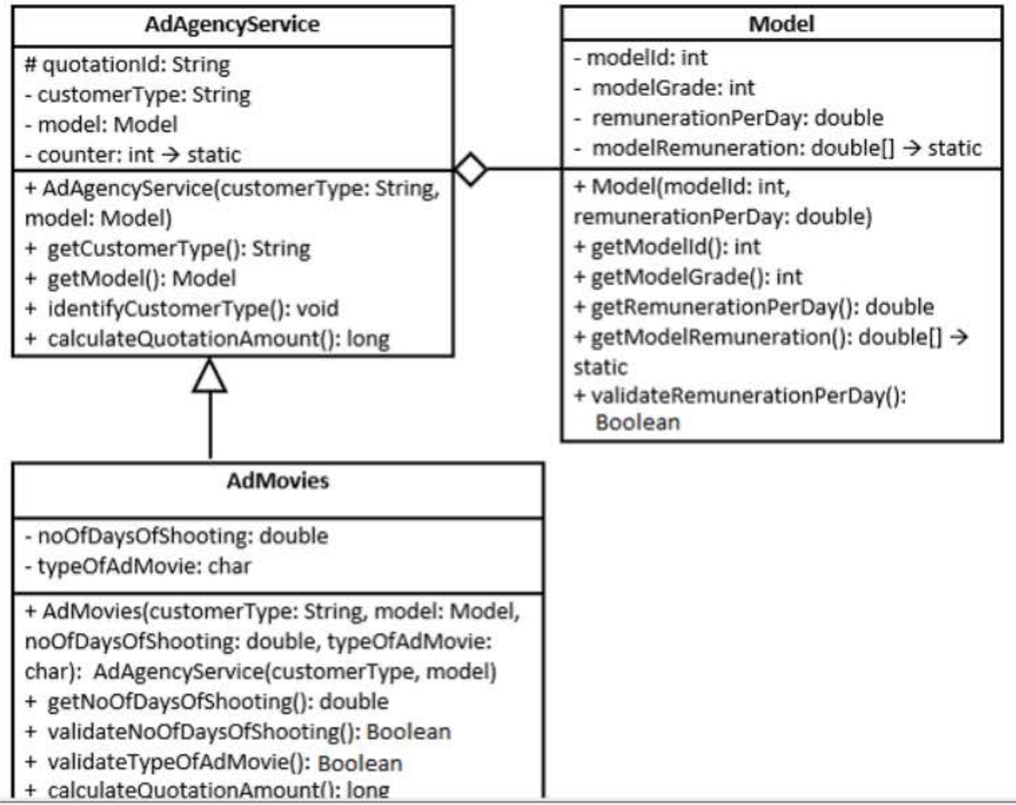
Problem Statement:

Description:

Advertise is an advertising movie shooting agency. It is popularly known for producing the creative advertisement movies. The automation of quotation generation system is depicted in the class diagram given below.

Implement the class diagram below to achieve the same.

Class Diagram:



Implementation Details:

Class Name	Implementation Details
AdAgencyService	Fully Implemented
AdMovies	Partially Implemented
Model	Partially Implemented

Model class:

modelRemuneration:

- This is a static array containing the following values-

modelRemuneration	{25000.0, 30000.0, 45000.0, 57500.0, 100000.0, 150000.0}

Note: modelRemuneration is initialized and provided. Hence no need to define

Constructor:

- This method initializes **modelId, modelGrade** and **remunerationPerDay**
- Extract the first digit of **modelId** and set that digit as **modelGrade**

For Example: if the **modelId** is **4**3216 then the first digit, i.e. 4 would be assigned as the **modelGrade**

Assumption: modelId is a 5digit positive nonzero integer number, not more than 59999

Note: No need to validate assumption

validateRemunerationPerDay() :

validateRemunerationPerDay() :

- This method validates the **remunerationPerDay**(double)
- Identify the *minimumLimit* and *maximumLimit* to validate **remunerationPerDay** using **modelRemuneration** array and **modelGrade**
- In **modelRemuneration** array, considering **modelGrade** as position of *maximumLimit* of **remunerationPerDay**, and the *minimumLimit* is the value available in the previous position of the same array.

  For example: When the **modelGrade** is 2, *maximumLimit* is 45000.0 (the value at index 2 in **modelRemuneration** array) and m*inimumLimit* is 30000.0 (the value at index 1 in **modelRemuneration** array)
- If the value of **remunerationPerDay** is more than the *minimumLimit* and less than or equal to *maximumLimit*, then return **true**
- Otherwise return **false**

Example: If the **modelId** is 12345, **modelRemuneration** is {25000.0, 30000.0, 45000.0, 57500.0, 100000.0, 150000.0} and **remunerationPerDay** is 28000.0, then this method would return **true**

AdMovies class:

validateNoOfDaysOfShooting() :

- This method validates **noOfDaysOfShooting**(double)
- If the value of **noOfDaysOfShooting** is between 2.0 to 100.0 (inclusive of boundary values) return **true**
- Otherwise return **false**

For Example: If **noOfDaysOfShooting** is 10.0 days, this method would return **true**

validateTypeOfAdMovie():

- This method validates **typeOfAdMovie** (char)
- If the value of **typeOfAdMovie** is either 'H' or 'A' or 'L', return **true**
- Otherwise return **false**

- If the value of **typeOfAdMovie** is either 'H' or 'A' or 'L', return **true**
- Otherwise return **false**

Note: Perform **case-sensitive** comparison

For Example: If **typeOfAdMovie** is 'H', this method would return **true**

calculateQuotationAmount():

- This method calculates and returns the *quotationAmount* as per the logic given below
- Invoke **validateTypeOfAdMovie(), validateNoOfDaysOfShooting()** and **validateRemunerationPerDay()**

  Note: validateRemunerationPerDay() needs to be called from **Model** class
- If all the above methods return **true**,
  - Identify *budget*(int) and *serviceCharge*(double) in percentage based on **typeOfAdMovie** as provided in the below table:

typeOfAdMovie	*budget*	*serviceCharge(%)*
'H'	2500000	25.25
'A'	2000000	17.5
'L'	1500000	12.75

  - Identify *tax*(double) based on **customerType** as provided in the below table:

customerType	*tax*(%)
"Government"	0.0
"Public"	15.75
"Private"	20.25

Note: Perform **case insensitive** comparison

- Calculate *actualNumberofShootingDays*(long) as mentioned below:
  - If **noOfDaysOfShooting** has a fractional part set the value of *actualNumberofShootingDays* to lower integral part

    Example: if **noOfDaysOfShooting** is 10.50 then *actualNumberofShootingDays* would be 10.0

  Note: You may use **Math.ceil()**, **Math.floor()** and **Math.round()** methods of **java.lang.Math** package to implement the mathematical functions
- Calculate *totalBudget*(double) using the below logic,

  totalBudget = *budget* + **remunerationPerDay** * *actualNumberofShootingDays*

  Where **remunerationPerDay** is retrieved from the **Model** class
- Calculate *serviceCharge* and *tax*Amount on *totalBudget* using the identified *serviceCharge* percentage, and *tax* percentage.
- Add them to *totalBudget*
- Invoke the **calculateQuotationAmount()** method of **AdAgencyService** to get the *baseQuotationAmount*
- Add this *baseQuotationAmount* to *totalBudget*
- Set the *quotationAmount* with the value of *totalBudget*
- Return *quotationAmount*

- Otherwise, return -1L.

For Example: If **customerType** is "public", **typeOfAdMovie** is 'H' **noOfDaysOfShooting** is 10.0, **remunerationPerDay** is 35000.0 and **modelId** is 23534 then, *quotationAmount* would be **4019000** L

WISH YOU ALL THE BEST

Tester.java

```java
package mock;
import mock.AdMovies;
import mock.Model;

public class Tester
{

    public static void main(String[] args)
    {
        Model modelobj = new Model(23534,35000.0);
        AdMovies admoviesobj = new AdMovies("pubLIC",modelobj,10.0,'H');
        System.out.println("Amount: "+admoviesobj.calculateQuotationAmount());
    }

}
```

Model.java

```java
package mock;

//DO NOT MODIFY THE CODE PROVIDED TO YOU

public class Model {
    private int modelId;
    private int modelGrade;
    private double remunerationPerDay;
    private static double[] modelRemuneration = {25000.0,30000.0,45000.0,57500.0,100000.0,150000.0};

    //To Trainees
    public Model(int modelId,double remunerationPerDay) {
        //Implement your code here
        this.modelId = modelId;
        this.modelGrade = modelId/10000;
        this.remunerationPerDay = remunerationPerDay;
    }
    public int getModelId() {
        return modelId;
    }
    public int getModelGrade() {
        return modelGrade;
    }
    public double getRemunerationPerDay() {
        return remunerationPerDay;
    }
    public static double[] getModelRemuneration() {
        return Model.modelRemuneration;
    }

    //To Trainees
    public Boolean validateRemunerationPerDay() {
        //Implement your logic here
        if(remunerationPerDay > modelRemuneration[modelGrade-1] && remunerationPerDay <= modelRemuneration[modelGrade]){
            return true;
        }
        return false;
    }
}
```

AdAgencyService.java | AdMovies.java

```java
package mock;

//DO NOT MODIFY THE CODE PROVIDED TO YOU

public class AdMovies extends AdAgencyService{
    private double noOfDaysOfShooting;
    private char typeOfAdMovie;

    public AdMovies(String customerType, Model model,double noOfDaysOfShooting,char typeOfAdMovie) {
        super(customerType, model);
        this.noOfDaysOfShooting = noOfDaysOfShooting;
        this.typeOfAdMovie = typeOfAdMovie;
    }


    public double getNoOfDaysOfShooting() {
        return noOfDaysOfShooting;
    }

    //To Trainees
    public Boolean validateNoOfDaysOfShooting() {
        //Implement your logic here
        if (noOfDaysOfShooting >= 2.0 && noOfDaysOfShooting <= 100.0){
            return true;
        }
        return false;

    }

    //To Trainees
    public Boolean validateTypeOfAdMovie() {
        //Implement your logic here
        if(typeOfAdMovie == 'H' || typeOfAdMovie == 'A' || typeOfAdMovie == 'L'){
            return true;
        }
        return false;

    }

    //To Trainees
    @Override
    public long calculateQuotationAmount(){
        int budget = 0;
```

AdAgencyService.java | AdMovies.java

```
39
40      //To Trainees
41      @Override
42      public long calculateQuotationAmount(){
43          int budget = 0;
44          long actualNumberOfShootingDays = (long) Math.floor(noOfDaysOfShooting);
45          double totalBudget;
46          double serviceCharge = 0, tax = 0;
47          long baseQuotationAmount =  super.calculateQuotationAmount();
48          long quotationAmount;
49          if(this.getModel().validateRemunerationPerDay() && validateNoOfDaysOfShooting() && validateTypeOfAdMovie()){
50              if(typeOfAdMovie == 'H'){
51                  budget = 2500000;
52                  serviceCharge = 25.25;
53              }
54              else if (typeOfAdMovie == 'A') {
55                  budget = 2000000;
56                  serviceCharge = 17.5;
57              }
58              else if (typeOfAdMovie == 'L') {
59                  budget = 1500000;
60                  serviceCharge = 12.75;
61              }
62              if(this.getCustomerType().equalsIgnoreCase("government")){
63                  tax = 0.0;
64              }
65              else if (this.getCustomerType().equalsIgnoreCase("public")) {
66                  tax = 15.75;
67              }
68              else if (this.getCustomerType().equalsIgnoreCase("private")) {
69                  tax = 20.25;
70              }
71              totalBudget = budget + (this.getModel().getRemunerationPerDay() * actualNumberOfShootingDays);
72              totalBudget = totalBudget + ((totalBudget/100)*serviceCharge) + ((totalBudget/100)*tax);
73              quotationAmount = (long) totalBudget + baseQuotationAmount;
74              return quotationAmount;
75          }
76
77          //Implement your code and change the return statement accordingly
78          return -1L;
79      }
80  }
81
```

AdAgencyService.java

```java
package mock;

//DO NOT MODIFY THE CODE PROVIDED TO YOU

public class AdAgencyService {
    protected String quotationId;
    private String customerType;
    private Model model;
    private static int counter = 1001;


    public AdAgencyService(String customerType,Model model) {
        this.customerType = customerType;
        this.model = model;
        identifyCustomerType();
        quotationId = getCustomerType().substring(0, 3).toUpperCase()+counter;
    }

    public String getCustomerType() {
        return customerType;
    }

    public Model getModel() {
        return model;
    }

    public void identifyCustomerType() {
        if(!(getCustomerType().equalsIgnoreCase("government") || getCustomerType().equalsIgnoreCase("public")
                || getCustomerType().equalsIgnoreCase("private") )) {
            customerType = "Private";
        }
    }

    public long calculateQuotationAmount()
    {
        long a = 500L;
        return a;
    }

}
```