ECE25100 Object-Oriented Programming
Assignment 2

****
NOTE:

1. Programs *must* compile without error. If your program does not compile, you will lose **50%** of the points. Even if your program is "essentially" correct, you will still lose 50% of the points; there is no reason to turn in a program with syntax errors.
2. To get full credit for an assignment your program must completely solve the stated problem.
3. Your program should contain a reasonable amount of comments. At the very beginning of each file, there should be a comment containing the course number, your name, the assignment number, and the date. It is good practice to insert comments into the source code, as you do when programming in any other languages.
```
/*
Class: ECE25100 Object Oriented Programming
Instructor: Xiaoli Yang
Author: [Your Name]
Assignment: [No.]
File Name:
Date: [MM]/[DD]/[YY]
*/
```
4. You should follow the programming styles and guidelines set forth in class. In particular, use meaningful names for variables. For each coding question you should also write a test class and test your code thoroughly.
5. Compress all .java files into one zip file and name it with your full name firstname_lastname_assignment #.zip. Submit the zip file on blackboard.

****

Buyer Class:

Implement a class called **Buyer** that represents a client of a real estate agency. Each **Buyer** object should keep track of its own name, address, phone number, and the mls number of the home purchased (assume only one has been purchased). As a result, the **Buyer** class defines four **private** instance fields or variables: **name**, **address**, and **phoneNumber** (all of type **String**) and a **mlsNumber** which is an int.

Implement the following instance methods in the **Buyer** class:

- **public** methods used to access and **protected** methods used to modify the instance fields or variables, i.e, **get** and **set** methods, respectively.
- a **toString()** method that returns an instance of a **String** containing the **Buyer** object's name and address, e.g., "Laura Wade, 123 main street, Highland, IN".
- a **constructor** that accepts a **String** for the name of the **Buyer** object, a **String** for the address of the **Buyer** object, and a **String** for the phone number of the **Buyer** object.
- a **zero-argument constructor** (i.e., one that takes no arguments) that sets the **Buyer** object's name to "name unknown", the **Buyer** object's address to "address unknown", and the **Buyer** object's phone number to "phone number unknown".

- a **viewHome** method that takes one parameter: aHome of type **Home**. The former is the **Home** object being viewed by a potential buyer. The method adds 1 to the numberOfPotentialBuyers instance field or variable of the **Home** object. (See the **Home** class below.)

Home Class

Implement a class called **Home** that represents a home being sold by a real estate agency. Each **Home** object should keep track of its own location, model, year of building, number of potential buyers, address, mls number, price, and availability. As a result, the **Home** class defines eight **private** instance fields or variables. Three of these instance variables or fields - **location**, **model**, and **address** - are of type **String**. The **availability** is of type boolean and indicates whether or not the home was sold. Three of these instance variables or fields - **year, mlsNumber,** and **noOfPotentialBuyers** - are of type **int**, while the remaining instance variable or field, **price**, is of type **double**.

Implement the following instance methods in the **Home** class:

- **public** methods used to access and **protected** methods used to modify the instance fields or variables, i.e, **get** and **set** methods, respectively.
- a **toString()** method that returns an instance of a **String** containing the **Home** object's location, model, and address, e.g., "port de leau 2 bdrm condo at 123 45th street, Highland, IN".
- a **constructor** that accepts a **String** for the location of the **Home** object, a **String** for the model of the **Home** object, a **String** for the address of the **Home** object, an **int** for the year of building of the **Home** object, an **int** for the number of potential buyers of the **Home** object, and an **int** for the mls number of the **Home** object. The availability instance variable should be initialized to **true**.

RealEstateAgent Class

Implement a class called **RealEstateAgent** that represents a real-estate agent working at a real estate agency. Each **RealEstateAngent** object should keep track of its own name, employee number, rate of commission, and total commission earned. As a result, the **RealEstateAngent** class defines four **private** instance fields or variables. The instance variable or field **employee Number** is of type **String**, while the instance variables or fields **commissionRate**, and **totalCommissionEarned** are of type **double**. The remaining instance variable or field **name** is of type **String**.

The instance field or variable **commissionRate** is the percentage of a sale that is paid to the sales person. For example, if the commissionRate is set to 0.15 (i.e., 15 percent), then when the real estate agent sells a home worth $235,000, 15 percent of that amount (**commissionRate** * 235000) is the real estate agent's commission.

The instance field or variable **totalCommissionEarned** is the sum of all commissions earned by the real estate agent. Each time the real estate agent earns some commission by selling a home, it should be added to the totalCommissionEarned variable. (See the addCommission(..) method below.)

Implement the following instance methods in the **RealEstateAgent** class:

- **public** methods used to access and **protected** methods used to modify the instance fields or variables, i.e, **get** and **set** methods, respectively.
- a **toString()** method that returns an instance of a **String** containing the **RealEstateAgent** object's name and employeeNumber, e.g., "Nick Smith, employee number: 000689".
- a **constructor** that accepts a **String** for the name of the **RealEstateAgent** object, a **String** for the **employeeNumber** of the **RealEstateAgent** object, and a **double** for the commissionRate of the **RealEstateAgent** object. As well, this constructor sets the initial value of totalCommissionEarned to 150.00.
- a **sellHome** method that accepts a **Home** object and a **Buyer** object as parameters. The method checks the availability of the **Home** object (if the home is not available the method returns null), changes the **Home** object's availability to false, calculates the commission earned by the sale, and adds this commission to the totalCommissionEarned by using the **addCommission** method (described below). The method returns a String that contains:
    - location, model, and address of the **Home** object
    - the name and address of the **Buyer** object
    - the name and employeeNumber of the **SalesPerson** object
    - e.g., " port de leau 2 bdrm condo at 123 45th street, Highland, IN sold to Laura Wade, 123 main street, Highland, IN by Nick Smith, employee number: 000689"

  Don't forget to set the mls number in the Buyer object and availability in the Home object.

- an **addCommission** method that takes a parameter **commissionEarned** of type **double**, adds the **commissionEarned** to the **totalCommissionEarned**, and returns the new value for **totalCommissionEarned** (again as a **double**).

Testing

To test your classes, implement a class called **RealEstateAgency**. This class will have no instance fields or variables, nor any instance methods. Instead, it will contain several **static** test methods written to test your classes, along with a main method that runs the test methods.

Be sure to test your code thoroughly and pick meaningful test cases.

Here is an example of one such test case: (some additional System.out.println() methods should be added)

```
public static void test1(){

   Home  bdrm2Condo, brdm1Appartment;
   Buyer laura, celeste;
   RealEstateAgent nick;

   // Make some homes, buyers and a real estate agent
   bdrm2Condo = new Home("port de leau", "2 bdrm condo", "123 45th, Highland,
   Indiana", 1986, 5, 00175);
   bdrm2Condo.setPrice(85000.00);
   brdm1Appartment = new Home("Tanglewood", "1 bdrm appartment", "2217
   Tanglewood Dr, Hammond, Indiana", 1972, 0, 00232);
   brdm1Appartment.setPrice(69000.00);
   laura = new Buyer("Laura Wade ", "123 main street", "246-8866");
   celeste = new Buyer("Celeste Sun", "662 California Street", "555-3217");
   nick = new RealEstateAgent("Nick Smith", "000135", .025);

   // do some home viewing and selling
   System.out.println(bdrm2Condo);
   System.out.println(laura);
   System.out.println(celeste);

   laura.viewHome(brdm1Appartment);
   celeste.viewHome(bdrm2Condo);
   celeste.viewHome(brdm1Appartment);
   System.out.println(nick.sellHome(bdrm2Condo, laura));
   System.out.println(nick.sellHome(bdrm2Condo, celeste));
}
```