# Chapter 10.B Implementing and Using Classes

21. Recall that the Fraction class has private fields num and denom, but no accessor methods for them. denom is always a positive integer. Is it possible to add to Fraction the following method?

// Returns true if this fraction is greater than other;

// otherwise returns false.

public boolean isGreaterThan(Fraction other)

If yes, write this method; if not, explain why not.

[10-22] 22. The class Balloon has a constructor that takes four parameters: *x* and *y* coordinates of the center, radius, and the color of the balloon. The library class Color has a constructor that takes three int parameters, the red, green, and blue components of the color. Suppose the class Canvas has a method public void add(Balloon b). Write a statement that adds a Balloon object to the canvas object of the Canvas class. The balloon should have the center at (50, 100), radius 20, and the color made of 60 red, 35 green, and 15 blue.

[10-23] 23. A playing card has a rank represented by an integer from 1 to 13 and a suit represented by its name (a String).

(a) Write a class PlayingCard to represent a playing card. Design your class in accordance with the encapsulation principle. Write a constructor that takes two parameters: the values for the card’s rank and suit. Also write a no-args constructor that sets the card’s rank to 1 and suit to "spades". Provide accessors for the rank and suit fields.

(b) Write a copy constructor for the PlayingCard class.

(c) Let’s say card1 beats card2 if they have the same suit and card1 has a higher rank. Write the following method of the PlayingCard class:

// Returns true if this card beats other; otherwise returns

// false.

public boolean beats(PlayingCard other)

(d) Provide a reasonable toString method.

(e) Write a client class that tests PlayingCard’s two constructors (the two-parameter constructor and the no-args constructor) and the beats method.

[10-24] 24. A class EggBasket has a copy constructor and a method

public void move(EggBasket other)

which moves all eggs from other to this EggBasket, leaving other empty. Is it possible to write a method of a client class that takes two EggBaskets as parameters and returns a new EggBasket that combines the eggs from both baskets, leaving both original baskets unchanged? If possible, write such a method; if not, explain why not.

[10-25] 25.

The program

public class Test

{

public static void main(String[] args)

{

LicensePlateMachine machine1 = new LicensePlateMachine();

LicensePlateMachine machine2 = new LicensePlateMachine();

for (int k = 0; k < 3; k++)

System.out.println(machine1.getPlate());

for (int k = 0; k < 3; k++)

System.out.println(machine2.getPlate());

}

}

produces the following output:

100001

100002

100003

100004

100005

100006

Write a short LicensePlateMachine class.

[10-26] 26. A class Car has a method getTankCapacity, which returns the capacity of the car’s fuel tank. It also has a method getGasAmt, which returns the current volume of gas in the tank and a method addGas(double gallons), which adds a given amount of gas to the tank. Write a class GasPump with the following features:

* A static field that represents the total amount of sales for all pumps;
* A static accessor method that returns the total amount of sales for all pumps;
* A constructor that takes one double parameter: gas price per gallon;
* A private method getCost(double gallons) that returns the cost of a given amount of gas;
* A private method pump(double gallons) that “dispenses” the specified amount of gas, appropriately updates the totalSales field, and returns the cost of the gas pumped;
* Two overloaded fill methods: fill(Car car) that fills the car’s tank, and fill(Car car, double dollarLimit) that pumps the maximum amount of gas that does not overflow car’s tank and does not exceed dollarLimit. Both fill methods return the cost of the pumped gas.

Avoid unnecessary duplication of code. For example, pump should call getCost and both fill methods should call pump.