Defining Your Own Classes—Part 2

7.1 Returning an Object from a Method

1. What’s wrong with the following declaration?

```java
class Question {
    Person student;

    public void getStudent() {
        return student;
    }

    ...
}
```

*The return type for the getStudent method must be Person as in*

```java
public Person getStudent() {
    return student;
}
```

2. Define a Vehicle class. It has a data member owner of type Person. Include an accessor to retrieve the owner person and a mutator to set the owner.
Solutions to Chapter 7 Quick Check Questions

Answer:

class Vehicle {
    private Person owner;
    
    public Person getOwner() {
        return owner;
    }
    
    public void setOwner(Person person) {
        owner = person;
    }
}

7.2 The Reserved Word this

1. Write a single statement to express the following operations on fractions using the methods from the Fraction class:

   \[ f_5 = \frac{(f_1 + f_2)}{(f_3 - f_4)} \]

   Answer:

   ```java
   //assume f1, ..., f5 are declared and
   //assigned values
   f5 = f1.add(f2).divide(f3.minus(f4));
   ```

2. If the add method is defined thus

   ```java
   public void add(Fraction frac) {
       int a, b, c, d;
       a = this.getNumerator(); //get this fraction’s
       b = this.getDenominator(); //num and denom
       c = frac.getNumerator(); //get frac’s num
       d = frac.getDenominator(); //and denom
       setNumerator(a*b + c*b); //updates this
       setDenominator(b*d); //fraction’s num and denom
   }
   ```
why is it wrong to use the method as

\[ f_3 = f_1.add(f_2); \]

*Because this add method is a void method. The method does not return a Fraction object. Rather, it adds the argument Fraction object to the receiving Fraction object. The correct would be*

\[ f_1.add(f_2); \]

3. Write statements to assign the sum of fractions \( f_1 \) and \( f_2 \) to fraction \( f_3 \) using the add method defined in Quick Check question 2 above.

\[
\begin{align*}
&f_1.add(f_2); \\
f_3 = f_1;
\end{align*}
\]

*The Fraction object \( f_3 \) becomes the sum of \( f_1 \) and \( f_2 \), but the value of \( f_1 \) will change also. If we do not want \( f_1 \) to change (which probably is the case for most situations), then we write*

\[
\begin{align*}
&f_3 = \text{new Fraction}(f_1); \\
f_3.add(f_2);
\end{align*}
\]

### 7.3 Overloaded Methods and Constructors

1. Are there any conflicts in the following three constructors for \texttt{ClassX} to be valid?

\[
\begin{align*}
\text{public ClassX( int X ) } & \{ \\
& \ldots \\
\} \\
\text{public ClassX( float X ) } & \{ \\
& \ldots \\
\}
\end{align*}
\]
public ClassX( int Y ) {
    
    ...
}

Yes, the first and the third constructors have the same signature so they cannot be overloaded.

2. Define a Student class. A Student has a name. Define two constructors, one with no argument and another with the name as its argument. Initialize the name to a default value Unknown for the zero-argument constructor.

Answer:

class Student {
    private String name;
    
    public Student( ) {
        name = "Unknown"; // or this("Unknown");
    }
    
    public Student(String name) {
        this.name = name;
    }
    ...
}

3. Rewrite the following constructors, so the first one calls the second one:

public ClassOne(int alpha) {
    this.alpha = alpha;
    this.beta  = 0;
}

public ClassOne(int alpha, int beta) {
    this.alpha = alpha;
    this.beta  = beta;
}
public ClassOne(int alpha) {
    this(alpha, 0);
}

7.4 Class Variables and Methods

No Quick Check Questions.

7.5 Call-by-Value Parameter Passing

1. What is the name of the scheme used in Java to pass arguments to a method?

   Call-by-value (also called as Pass-by-value).

2. What is the output from the following code?

   class Question {
       private int one;

       public void myMethod(int one) {
           this.one = one;
           one = 12;
       }
   }

   class Test {
       public static void main(String[] arg) {
           int one = 30;

           Question q = new Question();
           q.myMethod(one);

           System.out.println(one);
       }
   }

   Answer:

   30
Solutions to Chapter 7 Quick Check Questions

7.6 Organizing Classes into a Package

No Quick Check Questions.

7.7 Using Javadoc Comments for Class Documentation

1. Add javadoc comments to the following class:

```java
class Instructor {
    private String name;

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }
}
```

Answer:

```java
/**
 * The class to model an instructor
 *
 * @author Dr. Caffeine
 */
class Instructor {

    /**
     * the name of this instructor */
    private String name;

    /**
     * Assigns the name to this instructor
     * @param name the name to assign
     */
    public void setName(String name) {
        this.name = name;
    }

    /**
     * Returns this instructor’s name
     * @return the name of this instructor
     */
    public String getName() {
```
2. What is the purpose of @author tag?

*The @author tag is used to record the author of the class.*

### 7.8 The Complete Fraction Class

*No Quick Check Questions*

### 7.9 Sample Development: Library Overdue Checker

*No Quick Check Questions*