**Section 8.2 Time Class Case Study**

• The public methods of a class are also known as the class’s public services orpublic interface (p. [316](http://proquest.safaribooksonline.com/9780133813036/ch08_html#page_316)). They present to the class’s clients a view of the services the class provides.

• A class’s private members are not accessible to its clients.

• String class static method format (p. [318](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec3_html#page_318)) is similar to methodSystem.out.printf except that format returns a formatted String rather than displaying it in a command window.

• All objects in Java have a toString method that returns a String representation of the object. Method toString is called implicitly when an object appears in code where a String is needed.

#### Section 8.3 Controlling Access to Members

• The access modifiers public and private control access to a class’s variables and methods.

• The primary purpose of public methods is to present to the class’s clients a view of the services the class provides. Clients need not be concerned with how the class accomplishes its tasks.

• A class’s private variables and private methods (i.e., its implementation details) are not accessible to its clients.

#### Section 8.4 Referring to the Current Object’s Members with the this Reference

• An instance method of an object implicitly uses keyword this (p. [322](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec3_html#page_322)) to refer to the object’s instance variables and other methods. Keyword this can also be used explicitly.

• The compiler produces a separate file with the .classextension for every compiled class.

• If a local variable has the same name as a class’s field, the local variable shadows the field. You can use thethis reference in a method to refer to the shadowed field explicitly.

#### Section 8.5 Time Class Case Study: Overloaded Constructors

• Overloaded constructors enable objects of a class to be initialized in different ways. The compiler differentiates overloaded constructors (p. [324](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec4_html#page_324)) by their signatures.

• To call one constructor of a class from another of the same class, you can use the this keyword followed by parentheses containing the constructor arguments. If used, such a constructor call must appear as the first statement in the constructor’s body.

#### Section 8.6 Default and No-Argument Constructors

• If no constructors are provided in a class, the compiler creates a default constructor.

• If a class declares constructors, the compiler will not create a default constructor. In this case, you must declare a no-argument constructor (p. [327](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec10_html#page_327)) if default initialization is required.

#### Section 8.7 Notes on Set and Get Methods

• Set methods are commonly called mutator methods (p.[331](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec7_html#page_331)) because they typically change a value. Getmethods are commonly called accessor methods (p.[331](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec7_html#page_331)) or query methods. A predicate method (p. [332](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec16_html#page_332)) tests whether a condition is true or false.

#### Section 8.8 Composition

• A class can have references to objects of other classes as members. This is called composition (p. [332](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec16_html#page_332)) and is sometimes referred to as a has-a relationship.

#### Section 8.9 enum Types

• All enum types (p. [335](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec9_html#page_335)) are reference types. An enumtype is declared with an enum declaration, which is a comma-separated list of enum constants. The declaration may optionally include other components of traditional classes, such as constructors, fields and methods.

• enum constants are implicitly final, because they declare constants that should not be modified.

• enum constants are implicitly static.

• Any attempt to create an object of an enum type with operator new results in a compilation error.

• enum constants can be used anywhere constants can be used, such as in the case labels of switch statements and to control enhanced for statements.

• Each enum constant in an enum declaration is optionally followed by arguments which are passed to the enumconstructor.

• For every enum, the compiler generates a staticmethod called values (p. [336](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec20_html#page_336)) that returns an array of the enum’s constants in the order in which they were declared.

• EnumSet static method range (p. [337](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec20_html#page_337)) receives the first and last enum constants in a range and returns anEnumSet that contains all the constants between these two constants, inclusive.

#### Section 8.10 Garbage Collection

• The Java Virtual Machine (JVM) performs automatic garbage collection (p. [338](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec10_html#page_338)) to reclaim the memory occupied by objects that are no longer in use. When there are no more references to an object, the object is eligible for garbage collection. The memory for such an object can be re-claimed when the JVM executes its garbage collector.

#### Section 8.11 static Class Members

• A static variable (p. [338](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec10_html#page_338)) represents classwide information that’s shared among the class’s objects.

• static variables have class scope. A class’s public static members can be accessed through a reference to any object of the class, or they can be accessed by qualifying the member name with the class name and a dot (.). Client code can access a class’s private static class members only through methods of the class.

• static class members exist as soon as the class is loaded into memory.

• A method declared static cannot access a class’s instance variables and instance methods, be cause astatic method can be called even when no objects of the class have been instantiated.

• The this reference cannot be used in a staticmethod.

#### Section 8.12 static Import

• A static import declaration (p. [342](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec26_html#page_342)) enables you to refer to imported static members without the class name and a dot (.). A single static import declaration imports one static member, and a static import on demand imports all static members of a class.

#### Section 8.13 final Instance Variables

• In the context of an app’s code, the principle of least privilege (p. [343](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec28_html#page_343)) states that code should be granted only the amount of privilege and access that it needs to accomplish its designated task.

• Keyword final specifies that a variable is not modifiable. Such variables must be initialized when they’re declared or by each of a class’s constructors.

#### Section 8.14 Package Access

• If no access modifier is specified for a method or variable when it’s declared in a class, the method or variable is considered to have package access (p. [344](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec13_html#page_344)).

#### Section 8.15 Using BigDecimal for Precise Monetary Calculations

• Any application that requires precise floating-point calculations without rounding errors—such as those in financial applications—should instead use classBigDecimal (package java.math; p. [346](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec15_html#page_346)).

• BigDecimal static method valueOf (p. [347](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec30_html#page_347)) with adouble argument returns a BigDecimal that represents the exact value specified.

• BigDecimal method add (p. [347](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec30_html#page_347)) adds its argumentBigDecimal to the BigDecimal on which the method is called and returns the result.

• BigDecimal provides the constants ONE (1), ZERO (0) and TEN (10).

• BigDecimal method pow (p. [347](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec30_html#page_347)) raises its first argument to the power specified in its second argument.

• BigDecimal method multiply (p. [347](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec30_html#page_347)) multiplies its argument BigDecimal with the BigDecimal on which the method is called and returns the result.

• Class NumberFormat (package java.text; p. [347](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec30_html#page_347)) provides capabilities for formatting numeric values as locale-specific Strings. The class’s static methodgetCurrencyInstance returns a pre configuredNumberFormat for local-specific currency values.NumberFormat method format performs the formatting.

• Locale-specific formatting is an important part of internationalization—the process of customizing your applications for users’ various locales and spoken languages.

• BigDecimal gives you control over how values are rounded—by default all calculations are exact and no rounding occurs. If you do not specify how to roundBigDecimal values and a given value cannot be represented exactly an ArithmeticException occurs.

• You can specify the rounding mode for BigDecimal by supplying a MathContext object (package java.math) to class BigDecimal’s constructor when you create aBigDecimal. You may also provide a MathContext to various BigDecimal methods that perform calculations. By default, each pre-configured MathContext uses so called “bankers rounding.”

• A BigDecimal’s scale is the number of digits to the right of its decimal point. If you need a Big Decimalrounded to a specific digit, you can call BigDecimalmethod setScale.

### Self-Review Exercise

[**8.1**](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec20_html#ch08ans1) Fill in the blanks in each of the following statements:

a) A(n) \_\_\_\_\_\_\_\_ imports all static members of a class.

b) String class static method \_\_\_\_\_\_\_\_ is similar to method System.out.printf, but re-turns a formattedString rather than displaying a String in a command window.

c) If a method contains a local variable with the same name as one of its class’s fields, the local variable \_\_\_\_\_\_\_\_ the field in that method’s scope.

d) The public methods of a class are also known as the class’s \_\_\_\_\_\_\_\_ or \_\_\_\_\_\_\_.

e) A(n) \_\_\_\_\_\_\_\_ declaration specifies one class to import.

f) If a class declares constructors, the compiler will not create a(n) \_\_\_\_\_\_\_.

g) An object’s \_\_\_\_\_\_\_\_ method is called implicitly when an object appears in code where a String is needed.

h) Get methods are commonly called \_\_\_\_\_\_\_\_ or \_\_\_\_\_\_\_.

i) A(n) \_\_\_\_\_\_\_\_ method tests whether a condition is true or false.

j) For every enum, the compiler generates a staticmethod called \_\_\_\_\_\_\_\_ that returns an array of theenum’s constants in the order in which they were declared.

k) Composition is sometimes referred to as a(n) \_\_\_\_\_\_\_\_ relationship.

l) A(n) \_\_\_\_\_\_\_\_ declaration contains a comma-separated list of constants.

m) A(n) \_\_\_\_\_\_\_\_ variable represents classwide information that’s shared by all the objects of the class.

n) A(n) \_\_\_\_\_\_\_\_ declaration imports one staticmember.

o) The \_\_\_\_\_\_\_\_ states that code should be granted only the amount of privilege and access that it needs to accomplish its designated task.

p) Keyword \_\_\_\_\_\_\_\_ specifies that a variable is not modifiable after initialiation in a declaration or constructor.

q) A(n) \_\_\_\_\_\_\_\_ declaration imports only the classes that the program uses from a particular package.

r) Set methods are commonly called \_\_\_\_\_\_\_\_ because they typically change a value.

s) Use class \_\_\_\_\_\_\_\_ to perform precise monetary calculations.

t) Use the \_\_\_\_\_\_\_\_ statement to indicate that a problem has occurred.

### Answers to Self-Review Exercise

[**8.1**](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec19_html#ch08que1)

a) static import on demand.

b) format.

c) shadows.

d) public services, public interface.

e) single-type-import.

f) default constructor.

g) toString.

h) accessor methods, query methods.

i) predicate.

j) values.

k) has-a.

l) enum.

m) static.

n) single static import.

o) principle of least privilege.

p) final.

q) type-import-on-demand.

r) mutator methods.

s) BigDecimal.

t) throw.

### Exercises

**8.2****(Based on**[***Section 8.14***](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec14_html#ch08lev1sec14)**)** Explain the notion of package access in Java. Explain the negative aspects of package access.

**8.3** What happens when a return type, even void, is specified for a constructor?

**8.4 (Rectangle Class)** Create a class Rectangle with attributes length and width, each of which defaults to 1. Provide methods that calculate the rectangle’s perimeter and area. It has set and get methods for both length and width. The set methods should verify that length and width are each floating-point numbers larger than 0.0 and less than 20.0. Write a program to test class Rectangle.

**8.5 (Modifying the Internal Data Representation of a Class)** It would be perfectly reasonable for the Time2 class of [Fig. 8.5](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec10_html#ch08fig05) to represent the time internally as the number of seconds since midnight rather than the three integer valueshour, minute and second. Clients could use the same publicmethods and get the same results. Modify the Time2 class of[Fig. 8.5](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec10_html#ch08fig05) to implement the time as the number of seconds since midnight and show that no change is visible to the clients of the class.

**8.6****(Savings Account Class)** Create classSavingsAccount. Use a static variableannualInterestRate to store the annual interest rate for all account holders. Each object of the class contains a privateinstance variable savingsBalance indicating the amount the saver currently has on deposit. Provide methodcalculateMonthlyInterest to calculate the monthly interest by multiplying the savingsBalance byannualInterestRate divided by 12—this interest should be added to savingsBalance. Provide a static methodmodifyInterestRate that sets the annualInterestRate to a new value. Write a program to test class SavingsAccount. Instantiate two savingsAccount objects, saver1 andsaver2, with balances of $2000.00 and $3000.00, respectively. Set annualInterestRate to 4%, then calculate the monthly interest for each of 12 months and print the new balances for both savers. Next, set the annualInterestRateto 5%, calculate the next month’s interest and print the new balances for both savers.

**8.7 (Enhancing Class** ***Time2*)** Modify class Time2 of [Fig. 8.5](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec10_html#ch08fig05)to include a tick method that increments the time stored in a Time2 object by one second. Provide methodincrementMinute to increment the minute by one and method incrementHour to increment the hour by one. Write a program that tests the tick method, the incrementMinutemethod and the incrementHour method to ensure that they work correctly. Be sure to test the following cases:

a) incrementing into the next minute,

b) incrementing into the next hour and

c) incrementing into the next day (i.e., 11:59:59 PM to 12:00:00 AM).

**8.8 (Enhancing Class** ***Date*)** Modify class Date of [Fig. 8.7](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec17_html#ch08fig07) to perform error checking on the initializer values for instance variables month, day and year (currently it validates only the month and day). Provide a method nextDay to increment the day by one. Write a program that tests method nextDay in a loop that prints the date during each iteration to illustrate that the method works correctly. Test the following cases:

a) incrementing into the next month and

b) incrementing into the next year.

**8.9** Rewrite the code in [Fig. 8.14](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec28_html#ch08fig14) to use a separate importdeclaration for each static member of class Math that’s used in the example.

**8.10** Write an enum type TrafficLight, whose constants (RED, GREEN, YELLOW) take one parameter—the duration of the light. Write a program to test the TrafficLight enum so that it displays the enum constants and their durations.

**8.11 (Complex Numbers)** Create a class called Complex for performing arithmetic with complex numbers. Complex numbers have the form

realPart + imaginaryPart \* i

where i is

Write a program to test your class. Use floating-point variables to represent the private data of the class. Provide a constructor that enables an object of this class to be initialized when it’s declared. Provide a no-argument constructor with default values in case no initializers are provided. Provide public methods that perform the following operations:

a) Add two Complex numbers: The real parts are added together and the imaginary parts are added together.

b) Subtract two Complex numbers: The real part of the right operand is subtracted from the real part of the left operand, and the imaginary part of the right operand is sub tracted from the imaginary part of the left operand.

c) Print Complex numbers in the form (realPart,imaginaryPart).

**8.12 (Date and Time Class)** Create class DateAndTime that combines the modified Time2 class of [Exercise 8.7](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec21_html#ch08que7) and the modified Date class of [Exercise 8.8](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec21_html#ch08que8). Modify methodincrementHour to call method nextDay if the time is incremented into the next day. Modify methods toStringand toUniversalString to output the date in addition to the time. Write a program to test the new class DateAndTime. Specifically, test incrementing the time to the next day.

**8.13 (Set of Integers)** Create class IntegerSet. EachIntegerSet object can hold integers in the range 0–100. The set is represented by an array of booleans. Array elementa[i] is true if integer i is in the set. Array element a[j] isfalse if integer j is not in the set. The no-argument constructor initializes the array to the “empty set” (i.e., allfalse values).

Provide the following methods: The static method unioncreates a set that’s the set-theoretic union of two existing sets (i.e., an element of the new set’s array is set to true if that element is true in either or both of the existing sets—otherwise, the new set’s element is set to false). Thestatic method intersection creates a set which is the set-theoretic intersection of two existing sets (i.e., an element of the new set’s array is set to false if that element is false in either or both of the existing sets—otherwise, the new set’s element is set to true). Method insertElement inserts a new integer k into a set (by setting a[k] to true). MethoddeleteElement deletes integer m (by setting a[m] to false). Method toString returns a String containing a set as a list of numbers separated by spaces. Include only those elements that are present in the set. Use --- to represent an empty set. Method isEqualTo determines whether two sets are equal. Write a program to test class IntegerSet. Instantiate several IntegerSet objects. Test that all your methods work properly.

**8.14 (Date Class)** Create class Date with the following capabilities:

a) Output the date in multiple formats, such as

MM/DD/YYYY  
June 14, 1992  
DDD YYYY

b) Use overloaded constructors to create Date objects initialized with dates of the formats in part (a). In the first case the constructor should receive three integer values. In the second case it should receive a Stringand two integer values. In the third case it should receive two integer values, the first of which represents the day number in the year. [Hint: To convert the String representation of the month to a numeric value, compare Strings using the equalsmethod. For example, if s1 and s2 are Strings, the method call s1.equals(s2) returns true if theStrings are identical and otherwise returns false.]

**8.15 (Rational Numbers)** Create a class called Rationalfor performing arithmetic with fractions. Write a program to test your class. Use integer variables to represent theprivate instance variables of the class—the numerator and the denominator. Provide a constructor that enables an object of this class to be initialized when it’s declared. The constructor should store the fraction in reduced form. The fraction

2/4

is equivalent to 1/2 and would be stored in the object as 1 in the numerator and 2 in the denominator. Provide a no-argument constructor with default values in case no initializers are provided. Provide public methods that perform each of the following operations:

a) Add two Rational numbers: The result of the addition should be stored in reduced form. Implement this as a static method.

b) Subtract two Rational numbers: The result of the subtraction should be stored in reduced form. Implement this as a static method.

c) Multiply two Rational numbers: The result of the multiplication should be stored in reduced form. Implement this as a static method.

d) Divide two Rational numbers: The result of the division should be stored in reduced form. Implement this as a static method.

e) Return a String representation of a Rationalnumber in the form a/b, where a is the numeratorand b is the denominator.

f) Return a String representation of a Rational number in floating-point format. (Consider providing formatting capabilities that enable the user of the class to specify the number of digits of precision to the right of the decimal point.)

**8.16 (Huge Integer Class)** Create a class HugeIntegerwhich uses a 40-element array of digits to store integers as large as 40 digits each. Provide methods parse, toString,add and subtract. Method parse should receive a String, extract each digit using method charAt and place the integer equivalent of each digit into the integer array. For comparingHugeInteger objects, provide the following methods:isEqualTo, isNotEqualTo, isGreaterThan, isLessThan,isGreaterThanOrEqualTo and isLessThanOrEqualTo. Each of these is a predicate method that returns true if the relationship holds between the two HugeInteger objects and returns false if the relationship does not hold. Provide a predicate method isZero. If you feel ambitious, also provide methods multiply, divide and remainder. [Note: Primitiveboolean values can be output as the word “true” or the word “false” with format specifier %b.]

**8.17 (Tic-Tac-Toe)** Create a class TicTacToe that will enable you to write a program to play Tic-Tac-Toe. The class contains a private 3-by-3 two-dimensional array. Use anenum type to represent the value in each cell of the array. The enum’s constants should be named X, O and EMPTY (for a position that does not contain an X or an O). The constructor should initialize the board elements to EMPTY. Allow two human players. Wherever the first player moves, place an Xin the specified square, and place an O wherever the second player moves. Each move must be to an empty square. After each move, determine whether the game has been won and whether it’s a draw. If you feel ambitious, modify your program so that the computer makes the moves for one of the players. Also, allow the player to specify whether he or she wants to go first or second. If you feel exceptionally ambitious, develop a program that will play three-dimensional Tic-Tac-Toe on a 4-by-4-by-4 board [Note: This is an extremely challenging project!].

**8.18 (*Account*** **Class with** ***BigDecimal balance*)** Rewrite the Account class of [Section 3.5](http://proquest.safaribooksonline.com/9780133813036/ch03lev1sec5_html#ch03lev1sec5) to store the balance as aBigDecimal object and to perform all calculations usingBigDecimals.

### Exercises

**8.2 (Based on**[***Section 8.14***](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec14_html#ch08lev1sec14)**)** Explain the notion of package access in Java. Explain the negative aspects of package access.

**8.3** What happens when a return type, even void, is specified for a constructor?

**8.4 (Rectangle Class)** Create a class Rectangle with attributes length and width, each of which defaults to 1. Provide methods that calculate the rectangle’s perimeter and area. It has set and get methods for both length and width. The set methods should verify that length and width are each floating-point numbers larger than 0.0 and less than 20.0. Write a program to test class Rectangle.

**8.5 (Modifying the Internal Data Representation of a Class)** It would be perfectly reasonable for the Time2 class of [Fig. 8.5](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec10_html#ch08fig05) to represent the time internally as the number of seconds since midnight rather than the three integer valueshour, minute and second. Clients could use the same publicmethods and get the same results. Modify the Time2 class of[Fig. 8.5](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec10_html#ch08fig05) to implement the time as the number of seconds since midnight and show that no change is visible to the clients of the class.

**8.6 (Savings Account Class)** Create classSavingsAccount. Use a static variableannualInterestRate to store the annual interest rate for all account holders. Each object of the class contains a privateinstance variable savingsBalance indicating the amount the saver currently has on deposit. Provide methodcalculateMonthlyInterest to calculate the monthly interest by multiplying the savingsBalance byannualInterestRate divided by 12—this interest should be added to savingsBalance. Provide a static methodmodifyInterestRate that sets the annualInterestRate to a new value. Write a program to test class SavingsAccount. Instantiate two savingsAccount objects, saver1 andsaver2, with balances of $2000.00 and $3000.00, respectively. Set annualInterestRate to 4%, then calculate the monthly interest for each of 12 months and print the new balances for both savers. Next, set the annualInterestRateto 5%, calculate the next month’s interest and print the new balances for both savers.

**8.7 (Enhancing Class** ***Time2*)** Modify class Time2 of [Fig. 8.5](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec10_html#ch08fig05)to include a tick method that increments the time stored in a Time2 object by one second. Provide methodincrementMinute to increment the minute by one and method incrementHour to increment the hour by one. Write a program that tests the tick method, the incrementMinutemethod and the incrementHour method to ensure that they work correctly. Be sure to test the following cases:

a) incrementing into the next minute,

b) incrementing into the next hour and

c) incrementing into the next day (i.e., 11:59:59 PM to 12:00:00 AM).

**8.8 (Enhancing Class** ***Date*)** Modify class Date of [Fig. 8.7](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec17_html#ch08fig07) to perform error checking on the initializer values for instance variables month, day and year (currently it validates only the month and day). Provide a method nextDay to increment the day by one. Write a program that tests method nextDay in a loop that prints the date during each iteration to illustrate that the method works correctly. Test the following cases:

a) incrementing into the next month and

b) incrementing into the next year.

**8.9** Rewrite the code in [Fig. 8.14](http://proquest.safaribooksonline.com/9780133813036/ch08lev2sec28_html#ch08fig14) to use a separate importdeclaration for each static member of class Math that’s used in the example.

**8.10** Write an enum type TrafficLight, whose constants (RED, GREEN, YELLOW) take one parameter—the duration of the light. Write a program to test the TrafficLight enum so that it displays the enum constants and their durations.

**8.11 (Complex Numbers)** Create a class called Complex for performing arithmetic with complex numbers. Complex numbers have the form

realPart + imaginaryPart \* i

where i is

Write a program to test your class. Use floating-point variables to represent the private data of the class. Provide a constructor that enables an object of this class to be initialized when it’s declared. Provide a no-argument constructor with default values in case no initializers are provided. Provide public methods that perform the following operations:

a) Add two Complex numbers: The real parts are added together and the imaginary parts are added together.

b) Subtract two Complex numbers: The real part of the right operand is subtracted from the real part of the left operand, and the imaginary part of the right operand is sub tracted from the imaginary part of the left operand.

c) Print Complex numbers in the form (realPart,imaginaryPart).

**8.12 (Date and Time Class)** Create class DateAndTime that combines the modified Time2 class of [Exercise 8.7](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec21_html#ch08que7) and the modified Date class of [Exercise 8.8](http://proquest.safaribooksonline.com/9780133813036/ch08lev1sec21_html#ch08que8). Modify methodincrementHour to call method nextDay if the time is incremented into the next day. Modify methods toStringand toUniversalString to output the date in addition to the time. Write a program to test the new class DateAndTime. Specifically, test incrementing the time to the next day.

**8.13 (Set of Integers)** Create class IntegerSet. EachIntegerSet object can hold integers in the range 0–100. The set is represented by an array of booleans. Array elementa[i] is true if integer i is in the set. Array element a[j] isfalse if integer j is not in the set. The no-argument constructor initializes the array to the “empty set” (i.e., allfalse values).

Provide the following methods: The static method unioncreates a set that’s the set-theoretic union of two existing sets (i.e., an element of the new set’s array is set to true if that element is true in either or both of the existing sets—otherwise, the new set’s element is set to false). Thestatic method intersection creates a set which is the set-theoretic intersection of two existing sets (i.e., an element of the new set’s array is set to false if that element is false in either or both of the existing sets—otherwise, the new set’s element is set to true). Method insertElement inserts a new integer k into a set (by setting a[k] to true). MethoddeleteElement deletes integer m (by setting a[m] to false). Method toString returns a String containing a set as a list of numbers separated by spaces. Include only those elements that are present in the set. Use --- to represent an empty set. Method isEqualTo determines whether two sets are equal. Write a program to test class IntegerSet. Instantiate several IntegerSet objects. Test that all your methods work properly.

**8.14 (Date Class)** Create class Date with the following capabilities:

a) Output the date in multiple formats, such as

MM/DD/YYYY  
June 14, 1992  
DDD YYYY

b) Use overloaded constructors to create Date objects initialized with dates of the formats in part (a). In the first case the constructor should receive three integer values. In the second case it should receive a Stringand two integer values. In the third case it should receive two integer values, the first of which represents the day number in the year. [Hint: To convert the String representation of the month to a numeric value, compare Strings using the equalsmethod. For example, if s1 and s2 are Strings, the method call s1.equals(s2) returns true if theStrings are identical and otherwise returns false.]

**8.15 (Rational Numbers)** Create a class called Rationalfor performing arithmetic with fractions. Write a program to test your class. Use integer variables to represent theprivate instance variables of the class—the numerator and the denominator. Provide a constructor that enables an object of this class to be initialized when it’s declared. The constructor should store the fraction in reduced form. The fraction

2/4

is equivalent to 1/2 and would be stored in the object as 1 in the numerator and 2 in the denominator. Provide a no-argument constructor with default values in case no initializers are provided. Provide public methods that perform each of the following operations:

a) Add two Rational numbers: The result of the addition should be stored in reduced form. Implement this as a static method.

b) Subtract two Rational numbers: The result of the subtraction should be stored in reduced form. Implement this as a static method.

c) Multiply two Rational numbers: The result of the multiplication should be stored in reduced form. Implement this as a static method.

d) Divide two Rational numbers: The result of the division should be stored in reduced form. Implement this as a static method.

e) Return a String representation of a Rationalnumber in the form a/b, where a is the numeratorand b is the denominator.

f) Return a String representation of a Rational number in floating-point format. (Consider providing formatting capabilities that enable the user of the class to specify the number of digits of precision to the right of the decimal point.)

**8.16 (Huge Integer Class)** Create a class HugeIntegerwhich uses a 40-element array of digits to store integers as large as 40 digits each. Provide methods parse, toString,add and subtract. Method parse should receive a String, extract each digit using method charAt and place the integer equivalent of each digit into the integer array. For comparingHugeInteger objects, provide the following methods:isEqualTo, isNotEqualTo, isGreaterThan, isLessThan,isGreaterThanOrEqualTo and isLessThanOrEqualTo. Each of these is a predicate method that returns true if the relationship holds between the two HugeInteger objects and returns false if the relationship does not hold. Provide a predicate method isZero. If you feel ambitious, also provide methods multiply, divide and remainder. [Note: Primitiveboolean values can be output as the word “true” or the word “false” with format specifier %b.]

**8.17 (Tic-Tac-Toe)** Create a class TicTacToe that will enable you to write a program to play Tic-Tac-Toe. The class contains a private 3-by-3 two-dimensional array. Use anenum type to represent the value in each cell of the array. The enum’s constants should be named X, O and EMPTY (for a position that does not contain an X or an O). The constructor should initialize the board elements to EMPTY. Allow two human players. Wherever the first player moves, place an Xin the specified square, and place an O wherever the second player moves. Each move must be to an empty square. After each move, determine whether the game has been won and whether it’s a draw. If you feel ambitious, modify your program so that the computer makes the moves for one of the players. Also, allow the player to specify whether he or she wants to go first or second. If you feel exceptionally ambitious, develop a program that will play three-dimensional Tic-Tac-Toe on a 4-by-4-by-4 board [Note: This is an extremely challenging project!].

**8.18 (*Account*** **Class with** ***BigDecimal balance*)** Rewrite the Account class of [Section 3.5](http://proquest.safaribooksonline.com/9780133813036/ch03lev1sec5_html#ch03lev1sec5) to store the balance as aBigDecimal object and to perform all calculations usingBigDecimals.