Exercises

8.2 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. The class has attributes length and width, each of which defaults to 1. It has methods that calculate the perimeter and the area of the rectangle. 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 to represent the time internally as the number of seconds since midnight rather than the three integer values hour, minute and second. Clients could use the same public methods and get the same results. Modify the Time2 class of Fig. 8.5 to implement the Time2 as the number of seconds since midnight and show that no change is visible to the clients of the class.

8.6 *(Enhancing Class Time2)* Modify class Time2 of Fig. 8.5 to include a tick method that increments the time stored in a Time2 object by one second. Provide method incrementMinute to increment the minute and method incrementHour to increment the hour. The Time2 object should always remain in a consistent state. Write a program that tests the tick method, the incrementMinute method 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).

8.7 *(Enhancing Class Date)* Modify class Date of Fig. 8.7 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. The Date object should always remain in a consistent state. Write a program that tests the nextDay method in a loop that prints the date during each iteration of the loop to illustrate that the nextDay method works correctly. Test the following cases:

a) incrementing into the next month and

b) incrementing into the next year.

8.8 *(Returning Error Indicators from Methods)* Modify the *set* methods in class Time2 of Fig. 8.5 to return appropriate error values if an attempt is made to set one of the instance variables hour, minute or second of an object of class Time to an invalid value. [*Hint:* Use boolean return types on each method.] Write a program that tests these new *set* methods and outputs error messages when incorrect values are supplied.

8.9 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.

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

*realPart + imaginaryPart * i*

where *i* is

$\sqrt{-1}$

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 is 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 subtracted from the imaginary part of the left operand.

c) Print Complex numbers in the form (a, b), where a is the real part and b is the imaginary part.

8.11 *(Date and Time Class)* Create class DateAndTime that combines the modified Time2 class of Exercise 8.6 and the modified Date class of Exercise 8.7. Modify method incrementHour to call method nextDay if the time is incremented into the next day. Modify methods toStandardString and 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.

8.12 *(Enhanced Rectangle Class)* Create a more sophisticated Rectangle class than the one you created in Exercise 8.4. This class stores only the Cartesian coordinates of the four corners of the rectangle. The constructor calls a *set* method that accepts four sets of coordinates and verifies that each of these is in the first quadrant with no single *x*- or *y*-coordinate larger than 20.0. The *set* method also verifies that the supplied coordinates specify a rectangle. Provide methods to calculate the length, width, perimeter and area. The length is the larger of the two dimensions. Include a predicate method isSquare which determines whether the rectangle is a square. Write a program to test class Rectangle.

8.13 *(Set of Integers)* Create class IntegerSet. Each IntegerSet object can hold integers in the range 0–100. The set is represented by an array of booleans. Array element a[i] is true if integer *i* is in the set. Array element a[j] is false if integer *j* is not in the set. The no-argument constructor initializes the Java array to the "empty set" (i.e., a set whose array representation contains all false values).

Provide the following methods: Method union creates a third set that is the set-theoretic union of two existing sets (i.e., an element of the third set's array is set to true if that element is true in either or both of the existing sets—otherwise, the element of the third set is set to false). Method intersection creates a third set which is the set-theoretic intersection of two existing sets (i.e., an element of the third set's array is set to false if that element is false in either or both of the existing sets—otherwise, the element of the third set is set to true). Method insertElement inserts a new integer *k* into a set (by setting a[k] to true). Method deleteElement deletes integer *m* (by setting a[m] to false). Method toSetString 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 `String` and 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 `equals` method. For example, if `s1` and `s2` are strings, the method call `s1.equals( s2 )` returns `true` if the strings are identical and otherwise returns `false`.]

8.15 *(Huge Integer Class)* Create a class `HugeInteger` which uses a 40-element array of digits to store integers as large as 40 digits each. Provide methods `input`, `output`, `add` and `subtract`. For comparing `HugeInteger` 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:* Primitive boolean values can be output as the word "true" or the word "false" with format specifier `%b`.]