Chapter 3 Introduction to Classes and Objects

Section 3.2 Instance Variables, *set* Methods and *get* Methods

3.2 Q1: Each class you create becomes a new ________ that can be used to declare variables and create objects.
a. package
b. instance
c. library
d. type.
ANS: d. type.

3.2 Q2: You can declare new classes as needed; this is one reason Java is known as a(n) ________ language.
a. portable
b. incremental
c. extensible
d. virtual
ANS: c. extensible.

Section 3.2.1 `Account` Class with an Instance Variable, a *set* Method and a *get* Method

3.2.1 Q1: Which of the following statements is *false*?
a. Each class declaration that begins with the access modifier `private` must be stored in a file that has the same name as the class and ends with the `.java` filename extension.
b. Every class declaration contains keyword `class` followed immediately by the class's name.
c. Class, method and variable names are identifiers.
d. An object has attributes that are implemented as instance variables and carried with it throughout its lifetime.
ANS: a. Each class declaration that begins with the access modifier `private` must be stored in a file that has the same name as the class and ends with the `.java` filename extension. `private` should be `public`.

3.2.1 Q2: Which of the following statements is *false*?
a. By convention class names begin with an uppercase letter, and method and variable names begin with a lowercase letter.
b. Instance variables exist before methods are called on an object, while the methods are executing and after the methods complete execution.
c. A class normally contains one or more methods that manipulate the instance variables that belong to particular objects of the class.
d. Instance variables can be declared anywhere inside a class.
ANS: d. Instance variables are declared inside a class declaration but *outside* the bodies of the class's method declarations.

3.2.1 Q3: Which of the following statements is *true*?
a. Each object (instance) of the class shares the class's instance variables.
b. Most instance-variable declarations are preceded with the keyword `public`, which is an access modifier.
c. Variables or methods declared with access modifier `private` are accessible only to methods of the class in which they're declared.
d. None of the above is true.
ANS: c. Variables or methods declared with access modifier `private` are accessible only to methods of the class in which they're declared.

3.2.1 Q4: When a method terminates, the values of its local variables are ________.
a. saved
b. copied
c. restored

d. lost
ANS: d. lost

3.2.1 Q5: Which of the following statements is *false*?
a. Variables declared in the body of a particular method are local variables and can be used only in that method.
b. A method's parameters are local variables of the method.
c. Every method's body is delimited by left and right braces (`{` and `}`).
d. Keyword `null` indicates that a method will perform a task but will not return any information.
ANS: d. Keyword `null` indicates that a method will perform a task but will not return any information. Actually, keyword `void` indicates that a method will perform a task but will not return any information.

3.2.1 Q6: Which of the following statements is *false*?
a. The method's return type specifies the type of data returned to a method's caller.
b. Empty parentheses following a method name indicate that the method does not require any parameters to perform its task.
c. When a method that specifies a return type other than `void` is called and completes its task, the method must return a result to its calling method
d. Classes often provide `public` methods to allow the class's clients to *set* or *get* private instance variables; the names of these methods must begin with *set* or *get*.
ANS: d. Classes often provide `public` methods to allow the class's clients to *set* or *get* private instance variables; the names of these methods must begin with *set* or *get*. Actually, the names of these methods need not begin with *set* or *get*, but this naming convention is recommended.

Section 3.2.2 `AccountTest` Class That Creates and Uses an Object of Class `Account`

3.2.2 Q1: A class that creates an object of another class, then calls the object's methods, is called a(n) ________ class.
a. object-oriented
b. inherited
c. caller
d. driver.
ANS: driver.

3.2.2 Q2: Which of the following statements is *false*?
a. Scanner method `next` reads characters until any white-space character is encountered, then returns the characters as a `String`.
b. To call a method of an object, follow the object name with a comma, the method name and a set of parentheses containing the method's arguments.
c. A class instance creation expression begins with keyword `new` and creates a new object.
d. A constructor is similar to a method but is called implicitly by the `new` operator to initialize an object's instance variables at the time the object is created.
ANS: b. To call a method of an object, follow the object name with a comma, the method name and a set of parentheses containing the method's arguments. To call a method of an object, follow the object name with a *dot separator* (.), the method name and a set of parentheses containing the method's arguments.

3.2.2 Q3: Which of the following statements is *true*?
a. Local variables are automatically initialized.
b. Every instance variable has a default initial value—a value provided by Java when you do not specify the instance variable's initial value.
c. The default value for an instance variable of type `String` is `void`.
d. The argument types in the method call must be identical to the types of the corresponding parameters in the method's declaration.
e. **ANS: b. Every instance variable has a default initial value—a value provided by Java when you do not specify the instance variable's initial value.**

Section 3.2.3 Compiling and Executing an App with Multiple Classes

3.2.3 Q1: Which of the following statements is *false*?

a. The `javac` command can compile multiple classes at once; simply list the source-code filenames after the command with each filename separated by a comma from the next.
b. If the directory containing the app includes only one app's files, you can compile all of its classes with the command `javac *.java`.
c. The asterisk (*) in `javac *.java` indicates that all files in the current directory ending with the filename extension "`.java`" should be compiled.
d. All of the above are true.

ANS: a. The `javac` command can compile multiple classes at once; simply list the source-code filenames after the command with each filename separated by a comma from the next. Actually, each file name should be separated by a space.

Section 3.2.4 `Account` UML Class Diagram with an Instance Variable and *set* and *get* Methods

3.2.4 Q1: Which of the following statements is *false*?

a. In the UML, each class is modeled in a class diagram as a rectangle with three compartments. The top one contains the class's name centered horizontally in boldface. The middle one contains the class's attributes, which correspond to instance variables in Java. The bottom one contains the class's operations, which correspond to methods and constructors in Java.
b. UML represents instance variables as an attribute name, followed by a colon and the type.
c. Private attributes are preceded by the keyword `private` in the UML.
d. The UML models operations by listing the operation name followed by a set of parentheses. A plus sign (+) in front of the operation name indicates that the operation is a public.

ANS: c. Private attributes are preceded by the keyword `private` in the UML. Actually, private attributes are preceded by a minus sign (-) in the UML.

3.2.4 Q2: Which of the following statements is *true*?

a. The UML models a parameter of an operation by listing the parameter name, followed by a colon and the parameter value between the parentheses after the operation name.
b. The UML indicates an operation's return type by placing a colon and the return value after the parentheses following the operation name.
c. UML class diagrams do not specify return types for operations that do not return values.
d. Declaring instance variables `public` is known as data hiding or information hiding.

ANS: c. UML class diagrams do not specify return types for operations that do not return values.

Section 3.2.5 Additional Notes on Class `AccountTest`

3.2.5 Q1: You must call most methods other than ________ explicitly to tell them to perform their tasks.

a. `public` methods
b. `main`
c. `static` methods
d. `private` methods

ANS: b. `main`.

3.2.5 Q2: A key part of enabling the JVM to locate and call method `main` to begin the app's execution is the ________ keyword, which indicates that `main` can be called without first creating an object of the class in which the method is declared.

a. `stable`
b. `private`
c. `static`
d. `public`

ANS: c. `static`.

3.2.5 Q3: Which of the following statements is *false*?

a. Most classes you'll use in Java programs must be imported explicitly.

b. There's a special relationship between classes that are compiled in the same directory. By default, such classes are considered to be in the same package—known as the default package.
c. Classes in the same package are implicitly imported into `main`.
d. An `import` declaration is not required when one class in a package uses another in the same package.
ANS: c. Classes in the same package are implicitly imported into `main`. Actually, classes in the same package are implicitly imported into the source-code files of other classes in that package.

3.2.5 Q4: An `import` declaration is not required if you always refer to a class with its ________ name, which includes its package name and class name.
a. compile-time
b. default package
c. paired
d. fully qualified name
ANS: d. fully qualified name.

Section 3.2.6 Software Engineering with `private` Instance Variables and `public` *set* and *get* Methods

3.2.6 Q1: Declaring instance variables ________ is known as data hiding or information hiding.
a. `secure`
b. `private`
c. `static`
d. `masked`
ANS: b. `private`.

Section 3.3 Primitive Types vs. Reference Types

3.3 Q1: Types in Java are divided into two categories. The primitive types are `boolean`, `byte`, `char`, `short`, `int`, `long`, `float` and `double`. All other types are ________ types.
a. `static`
b. reference
c. declared
d. source
ANS: b. reference

3.3 Q2: Which of the following statements is *false*?
a. A primitive-type variable can store exactly one value of its declared type at a time.
b. Primitive-type instance variables are initialized by default.
c. Variables of types `byte`, `char`, `short`, `int`, `long`, `float` and `double` are initialized to `0`.
d. Variables of type `boolean` are initialized to `true`.
ANS: d. Variables of type `boolean` are initialized to `true`. Actually, variables of type `boolean` are initialized to `false`.

3.3 Q3: Reference-type variables (called references) store ________ in memory.
a. the value of an object
b. a copy of an object
c. the location of an object
d. the size of an object
ANS: c. the location of an object

3.3 Q4: Which of the following statements is *false*?
a. A reference to an object is required to invoke an object's methods.
b. A primitive-type variable does not refer to an object.
c. Reference-type instance variables are initialized by default to the value `void`.
d. A primitive-type variable cannot be used to invoke a method.
ANS: c. Reference-type instance variables are initialized by default to the value `void`. Actually, Reference-type instance variables are initialized by default to the value `null`.

Section 3.4 `Account` Class: Initializing Objects with Constructors

3.4 Q1: Java requires a ________ call for every object that's created.
a. constructor
b. destructor
c. parameterless
d. parameterized
ANS: a. constructor.

Section 3.4.1 Declaring an `Account` Constructor for Custom Object Initialization

3.4.1 Q1: Which of the following statements is *true*?
a. Constructors can specify parameters and return types.
b. Constructors can specify parameters but not return types.
c. Constructors cannot specify parameters but can specify return types.
d. Constructors can specify neither parameters nor return types.
ANS: b. Constructors can specify parameters but not return types.

Section 3.4.2 Class `AccountTest`: Initializing Account Objects When They're Created

3.4.2 Q1: If a class does not define constructors, the compiler provides a default constructor with no parameters, and the class's instance variables are initialized to ________.
a. zero
b. `null`
c. their default values.
d. `false`
ANS: c. their default values.

3.4.2 Q2: Which of the following statements is *false*?
a. If a class does not define constructors, the compiler provides a default constructor with no parameters.
b. If you declare a constructor for a class, the compiler will not create a default constructor for that class.
c. The UML models constructors in the third compartment of a class diagram.
d. To distinguish a constructor from a class's operations, the UML places the word "constructor" between double quotes before the constructor's name.
ANS: d. To distinguish a constructor from a class's operations, the UML places the word "constructor" between double quotes before the constructor's name. Actually, the UML places the word "constructor" between guillemets (« and ») before the constructor's name.

Section 3.5 `Account` Class with a Balance: Floating-Point Numbers

3.5 Q1: Which of the following statements is *false*?
a. A floating-point number is a number with a decimal point.
b. Java provides two primitive types for storing floating-point numbers in memory—`float` and `double`.
c. Variables of type `float` represent single-precision floating-point numbers and have seven significant digits.
d. Variables of type `double` represent double-precision floating-point numbers; these require twice as much memory as `float` variables and provide 14 significant digits.
ANS: Variables of type `double` represent double-precision floating-point numbers; these require twice as much memory as `float` variables and provide 14 significant digits. Actually, variables of type `double` provide 15 significant digits.

3.5 Q2: Floating-point literals are of type ________ by default.
a. `float`
b. `double`
c. `real`

d. `decimal`
ANS: b. `double`.

Section 3.5.1 `Account` Class with a `balance` Instance Variable of Type `double`

3.5.1 Q1: A __________ of a class called `myClass` is another class whose methods call the methods of `myClass`.
a. consumer
b. servant
c. caller
d. client
ANS: d. client

Section 3.5.2 `AccountTest` Class to Use Class `Account`

3.5.2 Q1: The format specifier ________ is used to output values of type `float` or `double`.
a. `%f`
b. `%d`
c. `%fd`
d. `%r`
ANS: a. `%f`

3.5.2 Q2: The format specifier `%.2f` specifies that two digits of precision should be output ________ in the floating-point number.
a. to the left of the decimal point
b. centered
c. to the right of the decimal point
d. None of the above.
ANS: c. to the right of the decimal point.

Section 3.6 (Optional) GUI and Graphics Case Study: Using Dialog Boxes

3.6 Q1: Which class provides prebuilt dialog boxes that enable programs to display windows containing messages (such windows are called *message dialogs*)?
a. `JDialogBox`.
b. `JPrebuiltDialog`.
c. `JMessageDialog`.
d. `JOptionPane`.
ANS: d. `JOptionPane`.

3.6 Q2: Which of the following statements is *false*?
a. You must create an object of class `JOptionPane` to use its `static` method `showMessageDialog`.
b. `JOptionPane` method `showInputDialog` displays an input dialog containing a prompt and a field (known as a text field) in which a user can enter text.
c. `String` method `format` works like method `System.out.printf`, except that `format` returns the formatted `String` rather than displaying it in a command window.
d. An input dialog allows the user to enter data into a program.
ANS: a. You must create an object of class `JOptionPane` to use its method `showMessageDialog`.