Glossary

A

absolute path—a complete file path that does not require any other information to locate a file on a system.

abstract class—a class from which no concrete objects can be instantiated, but which can serve as a basis for inheritance. Abstract classes usually have one or more empty abstract methods. Contrast with *concrete class*.

abstract data type—a type whose implementation is hidden and accessed through its public methods.

abstract method—a method declared with the keyword `abstract`; it is a method with no body that must be implemented in a subclass.

abstraction—the programming feature that allows a method name to encapsulate multiple statements.

accelerator—a key combination that causes a menu item to be chosen, whether or not the menu item is visible.

access modifier—an access specifier.

access specifier—defines the circumstances under which a class can be accessed and the other classes that have the right to use a class.

accessor methods—methods that return information about an object.

accumulating—the process of repeatedly increasing a value by some amount to produce a total.

action key—a keyboard key that does not generate a character.

`actionPerformed(ActionEvent e)` method—a method that defines the actions that occur in response to an event.

actual parameters—the arguments in a method call. Contrast with *formal parameters*.

acyclic gradient—a fill pattern in which a color shift occurs once between two points.

adapter class—a class that implements all the methods in an interface, providing an empty body for each method.

add and assign operator—an operator that alters the value of the operand on the left by adding the operand on the right to it; it is composed of a plus sign and an equal sign (`+=`).

`add()` method—a method that adds components to a container.

`addActionListener()` method—a method that tells a class to expect `ActionEvents`.

`addPoint()` method—a method that adds points to a `Polygon` object.

ad-hoc polymorphism—polymorphism that occurs when a single method name can

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

be used with a variety of data types because various implementations exist; it is another name for method overloading.

aggregation—a type of composition in which a class contains one or more members of another class that would continue to exist without the object that contains them.

algorithm—a process or set of steps that solves a problem.

Allman style—the indent style in which curly braces are aligned and each occupies its own line; it is named for Eric Allman, a programmer who popularized the style. Contrast with *K & R style*.

ambiguous—describes a situation in which the compiler cannot determine which method to use.

anonymous classes—nested, local classes that have no identifier.

anonymous object—an unnamed object.

append() method—a `StringBuilder` class method that adds characters to the end of a `StringBuilder` object.

applet—a Java program that is called from within another application, frequently a Web page.

Applet Viewer—a program that comes with the JDK and displays applets without using a Web browser.

appletviewer command—a command that starts the Applet Viewer.

application software—programs that perform tasks for users. Contrast with *system software*.

application-triggered painting—painting operations that occur when the internal state of a component has changed. Contrast with *system-triggered painting*.

arc—a portion of a circle.

architecturally neutral—describes the feature of Java that allows a program to run on any platform.

argument index—in a `printf()` statement, an integer that indicates the position of an argument in the argument list.

arguments—data items sent to methods in a method call.

arithmetic operators—operators used to perform calculations with values.

array—a named list of data items that all have the same type.

`ArrayList` class—a Java class that provides a dynamically resizable container that stores lists of objects.

`Arrays` class—a built-in Java class that contains many useful methods for manipulating arrays, such as methods to search, fill, compare, and sort arrays.

ascending order—the order of objects arranged from lowest to highest value. See also *descending order*.

ascent—one of three measures of a `Font`'s height; it is the height of an uppercase character from a baseline to the top of the character. See also *leading* and *descent*.

ASCII—an acronym for American Standard Code for Information Interchange, a character set widely used to represent computer data.

`assert` statement—a statement that creates an assertion.

assertion—a Java language feature that can help detect logic errors and debug a program.

assignment—the act of providing a value for a variable.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

assignment operator—the equal sign (=); any value to the right of the equal sign is assigned to the variable on the left of the equal sign.

associativity—describes the order in which operands are used with operators.

at run time—describes the period of time during which a program executes.

attributes—the characteristics that define an object as part of a class.

B

back buffer—the offscreen image during double buffering.

base class—a class that is used as a basis for inheritance.

`BasicStroke`—a class that defines line types and implements the `Stroke` interface.

batch processing—processing that involves performing the same tasks with many records, one after the other.

binary files—files that contain data that has not been encoded as text; their contents are in binary format.

binary numbering system—a numbering system based on two digits, 0 and 1, in which each column represents a value two times higher than the column to its right.

binary operators—operators that require two operands.

bit—a binary digit, 0 or 1, used to represent computerized values.

black box—a device that can be used solely in terms of input and output without regard to how it works internally.

blank `final`—a `final` variable that has not yet been assigned a value.

block—the code between a pair of curly braces.

block comments—comments that start with a forward slash and an asterisk (/*) and end with an asterisk and a forward slash (*/). Block comments can appear on a line by themselves, on a line before executable code, or on a line after executable code. Block comments also can extend across as many lines as needed. Contrast with *line comments*.

block line transfer or **blitting**—the act of copying contents from one surface to another.

Boolean values—true or false values; every computer decision results in a Boolean value.

`boolean` variable—a variable of the `boolean` data type that can hold only one of two values—`true` or `false`.

`BorderLayout`—a layout manager that divides a container into five regions.

`BorderLayout` manager—the default manager class for all content panes.

`BoxLayout` manager—a layout manager that allows multiple components to be laid out either vertically or horizontally. The components do not wrap, so a vertical arrangement of components, for example, stays vertically arranged when the frame is resized.

bubble sort—a type of sort in which pairs of items are compared and, if necessary, swapped so that the smallest items "bubble" to the top of the list, eventually creating a sorted list.

buffer—a memory location that holds data temporarily—for example, during input and output operations.

`ButtonGroup`—a UI component that groups several components, such as `JCheckBoxes`, so a user can select only one at a time.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

byte—the data type that holds very small integers, from –128 to 127.

bytecode—programming statements that have been compiled into binary format.

C

call a procedure—to invoke a method.

call stack—the memory location where the computer stores the list of method locations to which the system must return.

called method—a term used to describe the relationship between two methods; a method that is invoked by another.

calling method—a term used to describe the relationship between two methods; a method that invokes another.

camel casing—a naming style in which an identifier begins with a lowercase letter and subsequent words within the identifier are capitalized. Contrast with *Pascal casing*.

capacity—an attribute of an `ArrayList` whose value is the number of items it can hold without having to increase its size. Also, with a `StringBuilder` object, the actual length of the buffer, as opposed to that of the string contained in the buffer.

`capacity()` method—a `StringBuilder` class method that returns the actual length, or capacity, of the `StringBuilder` object.

`CardLayout` manager—a layout manager that generates a stack of containers or components, one on top of another.

cast operator—an operator that performs an explicit type conversion; it is created by placing the desired result type in parentheses before the expression to be converted.

`catch` block—a segment of code that can handle an exception that might be thrown by the `try` block that precedes it.

catch or specify requirement—the Java rule that checked exceptions require catching or declaration.

`char`—the data type that holds any single character.

character—any letter, number, or special symbol (such as a punctuation mark) that makes up data.

`Character` class—a class whose instances can hold a single character value. This class also defines methods that can manipulate or inspect single-character data.

`charAt()` method—a `String` and `StringBuilder` class method that requires an integer argument that indicates the position of the character that the method returns.

checked exceptions—exceptions that a programmer should plan for and from which a program should be able to recover. Contrast with *unchecked exceptions*.

child class—a derived class.

class—a group or collection of objects with common properties.

class body—the set of data items and methods between the curly braces that follow the class header.

class client—an application or class that instantiates objects of another class. See also *class user*.

class definition—a description of attributes and methods of objects instantiated from a class.

class diagram—a visual tool that provides an overview of a class. It consists of a rectangle divided into three sections—the top section contains the name of the class, the middle section contains the names and data types of the attributes, and the bottom section contains the methods.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

class methods—static methods that do not have a `this` reference (because they have no object associated with them).

class user—an application or class that instantiates objects of another prewritten class. See also *class client*.

class variables—static variables that are shared by every instantiation of a class.

class-level Javadoc comments—Javadoc comments that provide a description of a class and that should be placed above the code that declares a class.

clean build—a compilation that is created after deleting all previously compiled versions of a class.

`clearRect()` method—a method that draws a rectangle using the background color to create what appears to be an empty or "clear" rectangle.

client method—a method that calls another method.

close the file—to make a file no longer available to an application.

closer in scope—a term that describes the status of a local variable over others that it shadows.

collision—describes a class-naming conflict.

`Color` class—a class that defines built-in colors.

comes into scope—describes what happens to a variable when it is declared. Contrast with *goes out of scope*.

comma-separated values (CSV)—fields that are separated with a comma.

commands—program statements.

comment out—the technique of turning a program statement into a comment so the compiler will not execute its command.

`compareTo()` method—a `String` class method used to compare two `String`s; the method returns a number that describes the differences between the `String`s.

comparison operator—a relational operator.

compiler—a program that translates language statements into machine code. A compiler translates an entire program before executing it. Contrast with *interpreter*.

compile-time error—an error for which the compiler detects a violation of language syntax rules and is unable to translate the source code to machine code.

composition—describes the relationship between classes when an object of one class is a data field within another class. See also *has-a relationship*.

computer file—a collection of stored information in a computer system.

computer program—a set of instructions that tells a computer what to do; software.

computer simulations—programs that attempt to mimic real-world activities so that their processes can be improved or so that users can better understand how the real-world processes operate.

concatenated—describes values that are added onto the end of another value.

concatenation—the process of joining a variable to a string to create a longer string.

concrete class—a nonabstract class from which objects can be instantiated. Contrast with *abstract class*.

conditional operator—an operator that requires three expressions separated with a question mark and a colon; the operator is

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

used as an abbreviated version of the `if...else` structure.

confirm dialog box—a window that can be created using the `showConfirmDialog()` method in the `JOptionPane` class and that displays the options Yes, No, and Cancel.

console applications—programs that support character output to a computer screen in a DOS window.

constant—describes values that cannot be changed during the execution of an application.

constructor—a method that establishes an object.

consume—to retrieve and discard an entry without using it.

container—a type of component that holds other components so they can be treated as a single entity.

containment hierarchy—a tree of components that has a top-level container as its root (that is, at its uppermost level).

content pane—a component that contains all the visible components in a top-level container's user interface.

`copyArea()` method—a method that copies any rectangular area to a new location.

counter-controlled loop—a definite loop. Contrast with *event-controlled loop*.

counting—the process of continually incrementing a variable to keep track of the number of occurrences of some event.

crash—a premature, unexpected, and inelegant end to a program.

cyclic gradient—a fill pattern in which a shift between colors occurs repeatedly between two points.

D

data fields—data variables declared in a class outside of any method.

data files—files that consist of related records that contain facts and figures, such as employee numbers, names, and salaries.

data type—describes the type of data that can be stored in a variable, how much memory the item occupies, and what types of operations can be performed on the data.

dead code—unreachable statements.

debugging—the process of locating and repairing a program's error.

decimal numbering system—the numbering system based on 10 digits, 0 through 9, in which each column value is 10 times the value of the column to its right.

`DecimalFormat` class—a class that provides ways to easily convert numbers into strings, allowing leading and trailing zeros, prefixes and suffixes, grouping (thousands) separators, and the decimal separator to be used for formatting.

decision structure—a logical structure that involves choosing between alternative courses of action based on some value within a program.

declaration—another name for a method header; also, the statement that assigns a data type and identifier to a variable.

decrementing—the act of subtracting 1 from a variable.

default constructor—a constructor that requires no arguments.

default package—the unnamed package in which a class is placed if no package is specified.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

definite loop—a loop that executes a specific number of times; a counted loop. Contrast with *indefinite loop*.

derived class—a class that inherits from a base class.

descending order—the order of objects arranged from highest to lowest value. See also *ascending order*.

descent—one of three measures of a `Font`'s height; it measures the part of characters that "hang below" the baseline, such as the tails on the lowercase letters g and j. See also *ascent* and *leading*.

`destroy()` method—a method invoked in an applet when the user closes the browser or Applet Viewer.

development environment—a set of tools that helps programmers by providing such features as displaying a language's keywords in color.

dialog box—a GUI object resembling a window that displays messages.

direct access files—random access files.

directories—elements in a storage organization hierarchy. See also *folders*.

divide and assign operator—an operator that alters the value of the operand on the left by dividing the operand on the right into it; it is composed of a slash and an equal sign (/=).

documentation comments—comments that automatically generate well-formatted program documentation.

do-nothing loop—a loop that performs no actions other than looping.

`double`—a data type that can hold a floating-point value of up to 14 or 15 significant digits of accuracy. Contrast with *`float`*.

double buffering—the default buffering strategy in which `JPanel`s are drawn offscreen when they are updated and displayed only when complete.

`Double` class—a wrapper class that contains a simple `double` and useful methods to manipulate it.

double-precision floating-point number—a type of value that is stored in a `double`.

`do...while` loop—a loop that executes a loop body at least one time; it checks the loop control variable at the bottom of the loop after one repetition has occurred.

`draw3DRect()` method—a method that draws a rectangle that appears to have "shadowing" on two of its edges—the effect is that of a rectangle that is lit from the upper-left corner and slightly raised or slightly lowered.

`drawArc()` method—a method that draws an arc.

`drawLine()` method—a method that draws a straight line between two points on the screen.

`drawOval()` method—a method that draws an oval.

`drawPolygon()` method—a method that draws complex shapes.

`drawRect()` method—a method that draws the outline of a rectangle.

`drawRoundRect()` method—a method that draws rectangles with rounded corners.

`drawString()` method—a method that draws a `String` in a `JFrame` or other component.

dual-alternative `if`—a decision structure that takes one of two possible courses of action. Contrast with *single-alternative `if`*.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

dummy values—values the user enters that are not "real" data but just signals to stop data entry.

dynamic method binding—the ability of an application to select the correct subclass method when the program executes. See also *late method binding*.

dynamically resizable—describes an object whose size can change during program execution.

E

echoing the input—the act of repeating the user's entry as output so the user can visually confirm the entry's accuracy.

editable—describes a component that can accept keystrokes.

element—one variable or object in an array.

`else` clause—the part of an `if...else` statement that executes when the evaluated Boolean expression is false.

`else...if` clause—a format used in nested `if` statements in which each instance of `else` and its subsequent `if` are placed on the same line.

empty body—a block with no statements in it.

empty statement—a statement that contains only a semicolon.

encapsulation—the act of hiding data and methods within an object.

endcap styles—styles applied to the ends of lines that do not join with other lines; they include `CAP_BUTT`, `CAP_ROUND`, and `CAP_SQUARE`.

`endsWith()` method—a `String` class method that takes a `String` argument and returns `true` or `false` if a `String` object does or does not end with the specified argument.

enhanced `for` loop—a language construct that cycles through an array without specifying the starting and ending points for the loop control variable.

`enum` constants—the allowed values for an enumerated data type.

enumerated data type—a programmer-created data type with a fixed set of values.

`equals()` method—an `Object` class method that takes a single argument, which must be the same type as the type of the invoking object, and returns a Boolean value indicating whether two object references are equal. The method is overridden in the `String` class to evaluate the contents of two `String` objects to determine if they are equivalent.

`equalsIgnoreCase()` method—a `String` class method that ignores case when determining if two `String`s are equivalent.

equivalency operator—the operator composed of two equal signs that compares values and returns `true` if they are equal.

`Error` class—a class that represents more serious errors than the `Exception` class—those from which a program usually cannot recover.

escape sequence—a sequence that begins with a backslash followed by a character; the pair frequently represents a nonprinting character.

event—a result when a user takes action on a component.

event-controlled loop—an indefinite loop. Contrast with *counter-controlled loop*.

event-driven program—a program in which the user might initiate any number of events in any order.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

event handler—a method that executes because it is called automatically when an appropriate event occurs.

exception—in object-oriented terminology, an unexpected or error condition.

`Exception` class—a class comprising less serious errors than those from the `Error` class; the `Exception` class represents unusual conditions that arise while a program is running, and from which the program can recover.

exception handling—an object-oriented technique for managing errors.

exception specification—the practice of using the keyword `throws` followed by an `Exception` type in the method header. An exception specification is required when a method throws a checked `Exception` that it will not catch but will be caught by a different method.

executing—the act of carrying out a program statement or program.

explicit conversion—the data type transformation caused by using a cast operator.

extended—describes classes that have descended from another class.

`extends`—a keyword used to achieve inheritance in Java.

Extensible Hypertext Markup Language (XHTML)—an extension of HTML.

F

factory methods—methods that assist in object creation.

FAQs—frequently asked questions.

fault-tolerant—describes applications that are designed so that they continue to operate, possibly at a reduced level, when some part of the system fails.

field—a data variable declared in a class outside of any method. In reference to storage, a group of characters that has some meaning.

file channel—an object that is an avenue for reading and writing a file.

`Files` class—a class used to perform operations on files and directories, such as deleting them, determining their attributes, and creating input and output streams.

fill patterns—patterns that describe how drawing objects are filled in.

`fill3DRect()` method—a method that creates filled, three-dimensional rectangles.

`fillArc()` method—a method that creates a solid arc.

`fillOval()` method—a method that draws a solid, filled oval.

`fillPolygon()` method—a method that draws a solid shape.

`fillRect()` method—a method that draws a solid, or filled, rectangle.

`final`—the keyword that precedes named constants, that describes superclass methods that cannot be overridden in a subclass, and that describes classes in which all methods are final.

`finally` block—a block of code that executes at the end of a `try...catch` sequence.

fixed method binding—the opposite of dynamic method binding; it occurs when a subclass method is selected while the program compiles rather than while it is running. See also *static method binding*.

`float`—a data type that can hold a floating-point value of up to six or seven significant digits of accuracy. Contrast with ***`double`***.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

floating-point—describes a number that contains decimal positions.

floating-point division—the operation in which two values are divided and either or both are floating-point values.

flowchart—a tool that helps programmers plan a program's logic by writing the steps in diagram form, as a series of shapes connected by arrows.

`FlowLayout` manager—a layout manager that arranges components in rows across the width of a `Container`; when the current row is filled, additional `Components` are placed in new rows. By default, the components in each row are centered.

flushing—an operation to clear bytes that have been sent to a buffer for output but that have not yet been output to a hardware device.

folders—elements in a storage organization hierarchy. See also *directories*.

`Font` class—a Java class that holds typeface and size information.

`for` loop—a loop that can be used when a definite number of loop iterations is required.

`foreach` loop—the enhanced `for` loop.

formal parameters—the variables in a method declaration that accept the values from actual parameters. Contrast with *actual parameters*.

format specifier—in a `printf()` statement, a placeholder for a numeric value.

format string—in a `printf()` statement, a string of characters that includes optional text (that is displayed literally) and one or more format specifiers.

fragile—describes classes that are prone to errors.

fully qualified identifier—describes a filename that includes the entire hierarchy in which a class is stored.

fundamental classes—basic classes contained in the `java.lang` package that are automatically imported into every program. Contrast with *optional classes*.

G

garbage value—the unknown value stored in an uninitialized variable.

generic programming—a feature of languages that allows methods to be used safely with multiple data types.

`getAvailableFontFamilyNames()` method—a method that returns the names of all available fonts.

`getContentPane()` method—a method that returns a reference to a container's content pane.

`getDefaultToolkit()` method—a method that provides information about the system in use.

`getFontMetrics()` method—a method that returns a `FontMetrics` object that contains information about the leading, ascent, descent, and height of a font.

`getScreenResolution()` method—a method that returns the screen resolution on the current system.

`getScreenSize()` method—a method that returns the screen size as a `Dimension` object.

`getText()` method—a method that retrieves the `String` of text in a `Component`.

glass pane—a pane that resides above the content pane in a container. It can contain tool tips.

goes out of scope—describes what happens to a variable at the end of the block

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

in which it is declared. Contrast with *comes into scope*.

gradient fill—a gradual shift from one color at one coordinate point to a different color at a second coordinate point.

graphical user interfaces (GUIs)—environments that allow users to interact with a program in a graphical environment.

`Graphics` class—an abstract class that descends directly from `Object` and holds data about graphics operations and methods for drawing shapes, text, and images.

`Graphics2D` class—a class that provides tools for two-dimensional drawing.

`GridBagLayout` manager—a layout manager that allows the addition of `Components` to precise locations within the grid, as well as to indicate that specific `Components` should span multiple rows or columns within the grid.

`GridLayout` manager—a layout manager that divides a container surface into a grid.

H

hardware—the general term for computer equipment.

has-a relationship—a relationship based on composition.

hash code—a calculated number used to identify an object.

header—the first line of a method; its declaration.

heavyweight components—components that require interaction with the local operating system. Contrast with *lightweight components*.

height of a font—the sum of its leading, ascent, and descent.

hexadecimal numbering system—a numbering system based on 16 digits, 0 through F, in which each column represents a value 16 times higher than the column to its right.

high-level programming language—a language that uses a vocabulary of reasonable terms, such as "read," "write," or "add," instead of referencing the sequences of on and off switches that perform these tasks. Contrast with *low-level programming language*.

`</html>`—the tag that ends every HTML document.

`<html>`—the tag that begins every HTML document.

HTML, or **Hypertext Markup Language**—a simple language used to create Web pages for the Internet.

I

identifier—the name of a program component such as a class, object, or variable.

`if` clause—the part of an `if...else` statement that executes when the evaluated Boolean expression is true.

`if...else` statement—the statement that provides the mechanism to perform one action when a Boolean expression evaluates as `true`, and to perform a different action when a Boolean expression evaluates as `false`.

`if` statement—the single-alternative decision statement.

image—a likeness of a person or thing.

immutable—describes objects that cannot be changed.

implementation—the actions that execute within a method; the method body.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

implementation hiding—a principle of object-oriented programming that describes the encapsulation of method details within a class.

implicit conversion—the automatic transformation of one data type to another. Also called *promotion*.

`import` statement—a Java statement that allows access to a built-in Java class that is contained in a package.

inclusion polymorphism—the situation in which a single method implementation can be used with a variety of related objects because they are objects of subclasses of the parameter type. See also *pure polymorphism*.

incrementing—the act of adding 1 to a variable.

indefinite loop—a loop in which the final number of iterations is unknown. Contrast with *definite loop*.

`indexOf()` method—a `String` class method that determines whether a specific character occurs within a `String`. If it does, the method returns the position of the character; the first position of a `String` begins with zero. The return value is –1 if the character does not exist in the `String`.

infinite loop—a loop that never ends.

information hiding—the object-oriented programming principle used when creating private access for data fields; a class's private data can be changed or manipulated only by a class's own methods, and not by methods that belong to other classes.

inheritance—a mechanism that enables one class to inherit, or assume, both the behavior and the attributes of another class.

`init()` method—the first method called in any applet.

initialization—the act of making an assignment at the time of variable declaration.

inlining—an automatic process that optimizes performance in which calls to `final` methods are replaced with the expanded code of their definitions at each method call location.

inner block—a block contained in an outer block. See also *inside block*.

inner classes—nested classes that require an instance. See also *nonstatic member classes*.

inner loop—a loop that is contained entirely within another loop.

input dialog box—a GUI object that asks a question and provides a text field in which the user can enter a response.

`insert()` method—a `StringBuilder` class method that adds characters at a specific location within a `StringBuilder` object.

inside block—a block contained in an outside block. See also *inner block*.

instance—an existing object of a class.

instance methods—methods used with object instantiations. See also *nonstatic methods*.

instance variables—the data components of a class.

`instanceof` operator—an operator that determines whether an object that is the operand on the left is a member or descendant of the class that is the operand on the right.

instantiate—to create an instance; to create an object.

instantiation—an object; one tangible example of a class.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

`int`—the data type used to declare variables and constants that store integers in the range of –2,147,483,648 to +2,147,483,647.

integer—a whole number without decimal places.

`Integer` class—a wrapper class that contains a simple integer and useful methods to manipulate it.

integer division—the operation in which one integer value is divided by another; the result contains no fractional part.

interactive program—an application in which the user makes direct requests, as opposed to one in which input comes from a file.

interface—a construct similar to a class, except that all of its methods must be abstract and all of its data (if any) must be `static final`; it declares method headers but not the instructions within those methods. Also used to describe the part of a method that a client sees and uses—it includes the method's return type, name, and arguments.

interpreter—a program that translates language statements into machine code. An interpreter translates and executes one statement at a time. Contrast with *compiler*.

invoke—to call or execute a method.

is-a relationship—the relationship between an object and the class of which it is a member.

iteration—one loop execution.

J

`JApplet`—a `Swing` class that serves as the basis for inheritance for applets.

Java—a programming language developed by Sun Microsystems as an object-oriented language used both for general-purpose business applications and for interactive, World Wide Web–based Internet applications.

Java API—the application programming interface, a collection of information about how to use every prewritten Java class.

Java applications—stand-alone Java programs.

Java ARchive (JAR) file—a file that compresses the stored data.

Java Enterprise Edition (EE)—a Java edition that includes all of the classes in the Java SE, plus a number of classes that are more useful to programs running on servers.

Java Foundation Classes (JFC)—selected classes from the `java.awt` package, including `Swing` component classes.

Java interpreter—the program that checks bytecode and communicates with the operating system, executing the bytecode instructions line by line within the Java virtual machine.

Java Micro Edition (ME)—a Java platform that is used for small devices such as PDAs, cell phones, and other consumer appliances.

Java SE 7—the most recent version of Java. The full, official name is Java Platform, Standard Edition 7.

Java Virtual Machine (JVM)—a hypothetical (software-based) computer on which Java runs.

`java.lang`—the package that is implicitly imported into every Java program and that contains the fundamental classes.

Javadoc—a documentation generator that creates Application Programming Interface (API) documentation in Hypertext Markup Language (HTML) format from Java source code.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Javadoc comment—a special form of block comment that provides a standard way to document Java code.

Javadoc tag—a keyword within a comment that the Javadoc tool can process.

`JButton`—a `Component` the user can click with a mouse to make a selection.

`JCheckBox`—a UI component that consists of a label positioned beside a square; a user can click the square to display or remove a check mark. Usually, a `JCheckBox` is used to turn an option on or off.

`JComboBox`—a UI component that combines two features: a display area showing an option and a list box containing additional options. The display area contains either a button that a user can click or an editable field into which the user can type.

JDK—the Java Standard Edition Development Kit.

`JFrame`—a container with a title bar and border.

jGRASP—a development environment and source code editor.

`JLabel`—a built-in Java `Swing` class that holds displayable text.

`JOptionPane`—a Java class that produces dialog boxes.

`JPanel`—a plain, borderless surface that can hold lightweight UI components.

`JScrollPane`—a pane that provides scroll bars along the side or bottom, or both, so that the user can scroll initially invisible parts of the pane into view.

`JTextField`—a component into which a user can type a single line of text data.

juncture styles—styles applied to lines that join; they include `JOIN_MITER`, `JOIN_ROUND`, and `JOIN_BEVEL`.

K

K & R style—the indent style in which the opening brace follows the header line; it is named for Kernighan and Ritchie, who wrote the first book on the C programming language. Contrast with *Allman style*.

key field—the field in a record that makes the record unique from all others.

keyboard buffer—a small area of memory where keystrokes are stored before they are retrieved into a program. Also called the *type-ahead buffer*.

`KeyListener` interface—an interface that provides methods that respond to actions the user initiates from the keyboard. The `KeyListener` interface contains three methods—`keyPressed()`, `keyTyped()`, and `keyReleased()`.

keywords—the words that are part of a programming language.

L

late method binding—the ability of an application to select the correct subclass method when the program executes. See also *dynamic method binding*.

layout manager—a class that controls component positioning in a UI environment.

leading—one of three measures of a `Font`'s height; it is the amount of space between baselines. See also *ascent* and *descent*.

leaf menu item—a menu item that does not bring up another menu.

`length` field—a field that contains the number of elements in an array.

`length()` method—a `String` class method that returns the length of a `String`.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

lexicographical comparison—a comparison based on the integer Unicode values of characters.

library of classes—a folder that provides a convenient grouping for classes. See also *package*.

lightweight components—components written completely in Java that do not have to rely on the code written to run in the local operating system. Contrast with *heavyweight components*.

line comments—comments that start with two forward slashes (//) and continue to the end of the current line. Line comments can appear on a line by themselves or at the end of a line following executable code. Contrast with *block comments*.

listener—an object that is interested in and reacts to an event.

literal constant—a value that is taken literally at each use. See also *unnamed constant*.

literal string—a series of characters that appear exactly as entered. Any literal string in Java appears between double quotation marks.

local classes—nested classes that are local to a block of code.

local variable—a variable known only within the boundaries of a method.

logic—describes the order of program statements that produce correct results.

logic error—an error that occurs when a program compiles successfully but produces an error during execution.

logical AND operator—an operator used between Boolean expressions to determine whether both are `true`. The AND operator is written as two ampersands (`&&`).

logical OR operator—an operator used between Boolean expressions to determine whether either expression is `true`. The OR operator is written as two pipes (`||`).

long—the data type that holds very large integers, from –9,223,372,036,854,775,808 to 9,223,372,036,854,775,807.

look and feel—describes the default appearance and behavior of any user interface.

loop—a structure that allows repeated execution of a block of statements.

loop body—the block of statements that executes when the Boolean expression that controls the loop is `true`.

loop control variable—a variable whose value determines whether loop execution continues.

loop fusion—the technique of combining two loops into one.

lossless data compression—a set of rules that allows an exact replica of data to be reconstructed from a compressed version.

low-level programming language—a language that corresponds closely to a computer processor's circuitry. Contrast with *high-level programming language*. Compare with *machine language*.

lvalue—an expression that can appear on the left side of an assignment statement. Contrast with *rvalue*.

M

machine code—machine language.

machine language—circuitry-level language; a series of on and off switches. Compare with *low-level programming language*.

magic number—a value that does not have immediate, intuitive meaning or a number that cannot be explained without additional

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

knowledge. Unnamed constants are magic numbers.

matrix—a two-dimensional array.

member-level Javadoc comments—Javadoc comments that describe the fields, methods, and constructors of a class.

menu bar—a horizontal strip that is placed at the top of a container and that contains user options.

menus—lists of user options.

method—a program module that contains a series of statements that carry out a task.

method body—the set of statements between curly braces that follow the method header and carry out the method's actions.

method header—the declaration or first line of a method that contains information about how other methods interact with it.

method's type—the method's return type.

mission critical—describes any process that is crucial to an organization.

mnemonic—a key that causes an already visible menu item to be chosen.

modulus operator—the percent sign; when it is used with two integers, the result is an integer with the value of the remainder after division takes place. Also called the *remainder operator*; sometimes called just *mod*.

`MouseEvent`—the type of event generated by mouse manipulation.

`MouseInputListener` interface—an interface that implements all the methods in both the `MouseListener` and `MouseMotionListener` interfaces.

`MouseListener` interface—an interface that provides methods named `mousePressed()`, `mouseClicked()`, and `mouseReleased()` that are analogous to the keyboard event methods `keyPressed()`, `keyTyped()`, and `keyReleased()`.

`MouseMotionListener` interface—an interface that provides methods named `mouseDragged()` and `mouseMoved()` that detect the mouse being rolled or dragged across a component surface.

multidimensional arrays—arrays that contain two or more dimensions.

multimedia—describes the use of sound, images, graphics, and video in computer programs.

multiple inheritance—the capability to inherit from more than one class; Java does not support multiple inheritance.

multiply and assign operator—an operator that alters the value of the operand on the left by multiplying the operand on the right by it; it is composed of an asterisk and an equal sign.

mutator methods—methods that set field values.

N

named constant—a named memory location whose value cannot change during program execution; in Java, its declaration includes the keyword `final`.

NaN—a three-letter abbreviation for "Not a number."

nested—describes the relationship of statements, blocks, or classes when one contains the other.

nested classes—classes contained in other classes.

nested `if` statements—describes `if` statements when one is contained within the other.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

new operator—an operator that allocates the memory needed to hold an object.

nonstatic member classes—nested classes that require an instance. See also *inner classes*.

nonstatic methods—methods used with object instantiations. See also *instance methods*.

nonvolatile storage—storage that does not require power to retain information. Contrast with *volatile storage*.

NOT operator—the exclamation point (`!`); it negates the result of any Boolean expression.

`null` String—an empty `String` created by typing a set of quotes with nothing between them.

numeric constant—a number whose value is taken literally at each use.

O

object—an instance of a class.

`Object` class—a class defined in the `java.lang` package that is imported automati-cally into every Java program and that is the base class for all other Java classes.

object-oriented programming—a style of programming that involves creating classes, creating objects from those classes, and creating applications that use those objects. Contrast with *procedural programming*.

octothorpe—the pound sign.

one-dimensional array—an array that contains one column of values and whose elements are accessed using a single subscript. See also *single-dimensional array*.

open a file—the action that creates an object and associates a stream of bytes with it.

operand—a value used in an arithmetic statement.

operator precedence—the rules for the order in which parts of a mathematical expression are evaluated.

optional classes—classes that reside in packages that must be explicitly imported into programs. Contrast with *fundamental classes*.

out of bounds—describes a subscript that is not within the allowed range for an array.

outer block—a block that contains a nested block. See also *outside block*.

outer loop—a loop that contains another loop.

outside block—a block that contains a nested block. See also *outer block*.

overloading—describes using one term to indicate diverse meanings, or writing multiple methods with the same name but with different arguments.

override a method—the action in which a child class method takes precedence over one in the parent class that has the same name and argument list.

overrides—describes what a variable or method does to another with the same name when it takes precedence over the other variable.

P

package—a named collection or library of classes. See also *library of classes*.

`paint()` method—a method that runs when Java displays a screen and that is commonly overridden in programs that use graphics.

painting—the act of displaying or redisplaying a surface.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

parallel array—an array with the same number of elements as another, and for which the values in corresponding elements are related.

parameters—data items received by a method.

parent class—a base class.

parse—to break into component parts; the process the compiler uses to divide source code into meaningful portions for analysis.

`parseDouble()` method—a `Double` class method that takes a `String` argument and returns its `double` value.

`parseInt()` method—an `Integer` class method that takes a `String` argument and returns its integer value.

parsing—the process the compiler uses to divide source code into meaningful portions for analysis.

Pascal casing—the style of using an uppercase letter to begin an identifier and to start each new word in an identifier. Contrast with *camel casing*. Compare to *upper camel casing*.

passed by reference—describes a variable passed to a method when the address is passed to the method. Contrast with *passed by value*.

passed by value—describes a variable passed to a method when a copy is made in the receiving method. Contrast with *passed by reference*.

passing arguments—the act of sending arguments to a method.

path—the complete list of the disk drive plus the hierarchy of directories in which a file resides.

`Path` class—a Java class used to work with file information, such as location, size, creation date, and whether the file exists.

path delimiter—the character used to separate path components.

pattern `String`—an argument passed to the `DecimalFormat` constructor that is composed of symbols that determine what a formatted number looks like.

permanent storage devices—hardware storage devices that retain data even when power is lost.

pixels—the picture elements, or tiny dots of light, that make up the image on a video monitor.

point size argument—an argument to the `Font` constructor that is an integer that represents about 1/72 of an inch.

polymorphism—the feature of languages that allows the same word to be interpreted correctly in different situations based on the context; the act of using the same method name to indicate different implementations.

populating an array—the act of providing values for all of the elements in an array.

postfix `++` or the **postfix increment operator**—an operator that is composed by placing two plus signs to the right of a variable; it evaluates the variable, then adds 1 to it. Contrast with *prefix ++*.

posttest loop—a loop in which the loop control variable is tested after the loop body executes. Contrast with *pretest loop*.

preferred size—a `Component`'s default size.

prefix `++` or the **prefix increment operator**—an operator that is composed by placing two plus signs to the left of a variable; it adds 1 to the variable, then evaluates it. Contrast with *postfix ++*.

prefix and postfix decrement operators—operators that subtract 1 from a variable before and after evaluating it, respectively.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

pretest loop—a loop in which the loop control variable is tested before the loop body executes. Contrast with *posttest loop*.

primary key—a unique identifier for data within a database.

primary surface—the visible screen surface during double buffering.

priming read or **priming input**—the first input statement prior to a loop that will execute subsequent input statements for the same variable.

primitive type—a simple data type. Java's primitive types are `byte`, `short`, `int`, `long`, `float`, `double`, `char`, and `boolean`.

private access—refers to a field that no other classes can access.

procedural programming—a style of programming in which sets of operations are executed one after another in sequence. Contrast with *object-oriented programming*.

procedures—sets of operations performed by a computer program.

program—a set of written computer instructions.

program comments—nonexecuting statements added to a Java file for the purpose of documentation.

program files—files that store software instructions.

program statements—similar to English sentences; they carry out the tasks that programs perform.

programmer-defined data type—a type that is created by a programmer and not built into the language; a class.

promotion—an implicit conversion.

prompt—a message that requests and describes user input.

property—an instance variable, field, or attribute of a class.

`protected`—a Java keyword that provides an intermediate level of security between `public` and `private` access. `Protected` members are those that can be used by a class and its descendants.

pseudocode—a tool that helps programmers plan a program's logic by writing plain English statements.

pseudorandom—describes numbers that appear to be random but are the same set of numbers whenever the seed is the same.

pure polymorphism—the situation in which a single method implementation can be used with a variety of related objects because they are objects of subclasses of the parameter type. See also *inclusion polymorphism*.

R

ragged array—a two-dimensional array that has rows of different lengths.

random access files—files in which records can be located in any order.

random access memory (RAM)—temporary, volatile storage.

random number—a number whose value cannot be predicted.

range check—a series of statements that determine within which of a set of ranges a value falls.

range match—the process of comparing a value to the endpoints of numerical ranges to find a category in which the value belongs.

real-time—describes applications that require a record to be accessed immediately while a client is waiting.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

record—a collection of fields that contains data about an entity.

redeclare a variable—to attempt to declare a variable twice—an illegal action.

reference—a variable that holds a memory address.

reference to an object—the name for a memory address where the object is held.

reference types—objects that hold memory addresses.

`regionMatches()` method—a `String` class method that compares two `String` regions.

relational operator—an operator that compares two items; an expression that contains a relational operator has a Boolean value.

relative path—a path that depends on other path information to be complete.

remainder and assign operator—an operator that alters the value of the operand on the left by assigning the remainder when the left operand is divided by the right operand; it is composed of a percent sign and an equal sign (%=).

remainder operator—the percent sign; when it is used with two integers, the result is an integer with the value of the remainder after division takes place. Also called the *modulus operator*.

`remove()` method—a method that removes components from a container.

`repaint()` method—a method that executes when a window needs to be updated, such as when it contains new images.

`replace()` method—a `String` class method that replaces all occurrences of a specified character within a `String`.

rerender—to repaint or redisplay a drawing.

return a value—to send a data value from a called method back to the calling method.

`return` statement—a statement that ends a method and frequently sends a value from a called method back to the calling method.

return type—the type of data that, upon completion of a method, is sent back to its calling method.

robustness—describes the degree to which a system is resilient to stress, maintaining correct functioning.

root directory—the main directory of a storage device, outside any folders.

runtime error—an error that occurs when a program compiles successfully but does not execute.

runtime exceptions—unplanned exceptions that occur during a program's execution. The term is also used more specifically to describe members of the `RuntimeException` class.

rvalue—an expression that can appear only on the right side of an assignment statement. Contrast with *lvalue*.

S

sandbox—a safe area in which a program can run without causing harm to other areas of a system.

scalar—describes simple, primitive variables, such as `int`, `double`, or `char`.

scientific notation—a display format that more conveniently expresses large or small numeric values; a multidigit number is converted to a single-digit number and multiplied by 10 to a power.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

scope—in reference to a variable, the portion of a program within which the variable exists and can be referenced.

SDK—a software development kit, or a set of tools useful to programmers; the Java EE Development Kit.

searching an array—the process of comparing a value to a list of values in an array, looking for a match.

seed—a starting value.

seekable—describes a file channel in which operations can start at any specified position.

semantic errors—the type of errors that occur when a correct word is used in the wrong context in program code.

sequence structure—a logical structure in which one step follows another unconditionally.

sequential access file—a data file in which each record is stored in order, based on the value in some field.

`setCharAt()` method—a `StringBuilder` class method that changes a character at a specified position within a `StringBuilder` object.

`setEditable()` method—a method that changes the editable status of a `JTextField`.

`setEnabled()` method—a method that makes a component available or dimmed and unavailable.

`setFont()` method—a method that changes a `JLabel`'s font.

`setLength()` method—a `StringBuilder` class method that changes the length of the characters in the `String` in a `StringBuilder` object.

`setLocation()` method—a method that places a component at a specific location within a `JFrame`'s content pane.

`setStroke()` method—a Java 2D method that alters a stroke's width.

`setText()` method—a method that changes the text in a `Component` that has already been created.

`setToolTipText()` method—a method that defines the text to be displayed in a tool tip.

shadowing—the action that occurs when a local variable hides a variable with the same name that is further away in scope.

`short`—the data type that holds small integers, from –32,768 to 32,767.

short-circuit evaluation—describes the feature of the AND and OR operators in which evaluation is performed only as far as necessary to make a final decision.

`showInputDialog()` method—a method that creates an input dialog box.

signature—a method's name and the number, types, and order of arguments.

significant digits—refers to the mathematical accuracy of a value.

single-alternative `if`—a decision structure that performs an action, or not, based on one alternative. Contrast with *dual-alternative `if`*.

single-dimensional array—an array that contains one column of values and whose elements are accessed using a single subscript. See also *one-dimensional array*.

single-precision floating-point number—a type of value that is stored in a `float`.

software—the general term for computer programs.

sorting—the process of arranging a series of objects in some logical order.

source—a component on which an event is generated.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

source code—programming statements written in a high-level programming language.

stack trace history list, or more simply **stack trace**—a list that displays all the methods that were called during program execution.

standard arithmetic operators—operators that are used to perform common calculations.

standard input device—normally the keyboard.

standard output device—normally the monitor.

`start()` method—a method that executes after the `init()` method in an applet and executes again every time the applet becomes active after it has been inactive.

`startsWith()` method—a `String` class method that takes a `String` argument and returns `true` or `false` if a `String` object does or does not start with the specified argument, respectively.

state—the values of the attributes of an object.

`static`—a keyword that means a method is accessible and usable even though no objects of the class exist.

`static` member class—a type of nested class that has access to all `static` methods of its top-level class.

static method binding—the opposite of dynamic method binding; it occurs when a subclass method is selected while the program compiles rather than while it is running. See also *fixed method binding*.

`stop()` method—a method invoked in an applet when a user leaves a Web page (perhaps by minimizing a window or traveling to a different Web page).

stream—a pipeline or channel through which bytes flow into and out of an application.

`String` class—a class used to work with fixed-string data—that is, unchanging data composed of multiple characters.

`String` variable—a named object of the `String` class.

`StringBuilder` and `StringBuffer` classes—classes used to store and manipulate changeable data composed of multiple characters that are used as alternatives to the `String` class.

`stringWidth()` method—a `FontMetrics` class method that returns the integer width of a `String`.

stroke—a line-drawing feature in Java 2D that represents a single movement using a drawing tool similar to a pen or pencil.

strongly typed language—a language in which all variables must be declared before they can be used.

stub—a method that contains no statements; programmers create stubs as temporary placeholders during the program development process.

style argument—an argument to the `Font` constructor that applies an attribute to displayed text and is one of three values: `Font.PLAIN`, `Font.BOLD`, or `Font.ITALIC`.

subclass—a derived class.

subscript—an integer contained within square brackets that indicates one of an array's variables, or elements.

`substring()` method—a `String` class method that extracts part of a `String`.

subtract and assign operator—an operator that alters the value of the operand on the left by subtracting the operand on

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

the right from it; it is composed of a minus sign and an equal sign (–=).

subtype polymorphism—the ability of one method name to work appropriately for different subclasses of a parent class.

`super`—a Java keyword that always refers to a class's immediate superclass.

superclass—a base class.

`Swing` components—UI elements such as dialog boxes and buttons; their names usually begin with *J*.

`switch` statement—a statement that uses up to four keywords to test a single variable against a series of exact integer or character values. The keywords are `switch`, `case`, `break`, and `default`.

symbolic constant—a named constant.

syntactic salt—describes a language feature designed to make it harder to write bad code.

syntactic sugar—describes aspects of a computer language that make it "sweeter," or easier, for programmers to use.

syntax—the rules of a language.

syntax error—a programming error that occurs when a program contains typing errors or incorrect language use. A program containing syntax errors will not compile.

system software—the set of programs that manage the computer. Contrast with *application software*.

`System.out.printf()` method—a method used to format numeric values.

system-triggered painting—painting operations that occur when the system asks a component to render its contents. Contrast with *application-triggered painting*.

T

table—a two-dimensional array; a matrix.

tag attributes—arguments that promote activity or describe the features of an HTML tag.

tags—HTML commands.

ternary operator—an operator that needs three operands.

text files—files that contain data that can be read in a text editor because the data has been encoded using a scheme such as ASCII or Unicode.

`this` reference—a reference to an object that is passed to any object's nonstatic class method.

threads of execution—units of processing that are scheduled by an operating system and that can be used to create multiple paths of control during program execution.

`throw` statement—a statement that sends an `Exception` out of a block or a method so it can be handled elsewhere.

TOCTTOU bug—an error that occurs when changes take place from Time Of Check To Time Of Use.

token—a unit of data; the `Scanner` class separates input into tokens.

`toLowerCase()` method—a `String` class method that converts any `String` to its lowercase equivalent.

tool tips—popup windows that can help a user understand the purpose of components in an application; a tool tip appears when a user hovers a mouse pointer over the component.

top-level class—the containing class in nested classes.

top-level container—a container at the top of a containment hierarchy. The Java

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

top-level containers are `JFrame`, `JDialog`, and `JApplet`.

`toString()` method—an `Object` class method that converts an `Object` into a `String` that contains information about the `Object`. Also, a `String` class method that converts any object to a `String`.

`toUpperCase()` method—a `String` class method that converts any `String` to its uppercase equivalent.

`try` block—a block of code that a programmer acknowledges might generate an exception.

two-dimensional array—an array that contains two or more columns of values and whose elements are accessed using multiple subscripts. Contrast with *one-dimensional array*.

type-ahead buffer—the keyboard buffer.

type casting—an action that forces a value of one data type to be used as a value of another type.

type conversion—the process of converting one data type to another.

typeface argument—an argument to the `Font` constructor that is a `String` representing a font. Common fonts have names such as Arial, Century, Monospaced, and Times New Roman.

type-safe—describes a data type for which only appropriate behaviors are allowed.

type-wrapper classes—a method that can process primitive type values.

U

UI components—user interface components, such as buttons and text fields, with which the user can interact.

unary cast operator—a more complete name for the cast operator that performs explicit conversions.

unary operator—an operator that uses only one operand.

unchecked exceptions—exceptions that cannot reasonably be expected to be recovered from while a program is executing. Contrast with *checked exceptions*.

Unicode—an international system of character representation.

Unified Modeling Language (UML)—a graphical language used by programmers and analysts to describe classes and object-oriented processes.

unifying type—a single data type to which all operands in an expression are converted.

uninitialized variable—a variable that has been declared but that has not been assigned a value.

unnamed constant—a constant value that has no identifier associated with it. See also *literal constant*.

unreachable statements—statements that cannot be executed because the logical path can never encounter them; in some languages, including Java, an unreachable statement causes a compiler error. See also *dead code*.

upcast—to change an object to an object of a class higher in its inheritance hierarchy.

upper camel casing—Pascal casing.

V

validating data—the process of ensuring that a value falls within a specified range.

variable—a named memory location whose contents can be altered during program execution.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

variable declaration—a statement that reserves a named memory location.

viewport—the viewable area in a `JScrollPane`.

virtual classes—the name given to abstract classes in some other programming languages, such as C++.

virtual key codes—codes that represent keyboard keys that have been pressed.

virtual method calls—method calls in which the method used is determined when the program runs, because the type of the object used might not be known until the method executes. In Java, all instance method calls are virtual calls by default.

`void`—a keyword that, when used in a method header, indicates that the method does not return any value when it is called.

volatile storage—memory that requires power to retain information. Contrast with *nonvolatile storage*.

W

Web browser—a program that displays HTML documents on the screen.

`while` loop—a construct that executes a body of statements continually as long as the Boolean expression that controls entry into the loop continues to be `true`.

whitespace—any combination of nonprinting characters; for example, spaces, tabs, and carriage returns (blank lines).

wildcard symbol—a symbol used to indicate that it can be replaced by any set of characters. In Java, the wildcard symbol is an asterisk; in an `import` statement, the wildcard symbol represents all the classes in a package.

window decorations—the icons and buttons that are part of a window or frame.

windowed applications—programs that create a graphical user interface (GUI) with elements such as menus, toolbars, and dialog boxes.

wrapped—to be encompassed in another type.

wrapper—a class or object that is "wrapped around" a simpler element.

"write once, run anywhere" (WORA)—a slogan developed by Sun Microsystems to describe the ability of one Java program version to work correctly on multiple platforms.

X

x-axis—an imaginary horizontal line that indicates screen position.

x-coordinate—a position value that increases from left to right across a window.

XHTML—Extensible Hypertext Markup Language, an extension of HTML.

Y

y-axis—an imaginary vertical line that indicates screen position.

y-coordinate—a position value that increases from top to bottom across a window.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.

Copyright 2013 Cengage Learning. All Rights Reserved. May not be copied, scanned, or duplicated, in whole or in part. Due to electronic rights, some third party content may be suppressed from the eBook and/or eChapter(s). Editorial review has deemed that any suppressed content does not materially affect the overall learning experience. Cengage Learning reserves the right to remove additional content at any time if subsequent rights restrictions require it.