1. data type

**CORRECT**

the kind of data stored in each variable

1. translates a Java source file into a Java bytecode file

**CORRECT**

compiler

1. using the (+) sign to combine strings

**CORRECT**

concatenate strings

1. dot pitch

**CORRECT**

the amount of space between pixels, measured in millimeters

1. logic error

**CORRECT**

occurs when a program does not perform the way it was intended to

1. Conditional Expression ( ? :)

**THE ANSWER**

Evaluates an expression based on a condition (pg 103)

1. directive

**CORRECT**

import

1. result from errors in code construction, such as misspellings, wrong punctuation, etc.

**CORRECT**

syntax error

1. octa integer

**THE ANSWER**

075

1. var++, + and -, casting, !, \* / %, + - concaction, (See page 105)

**CORRECT**

operator precedence

1. If you try to store a value in a data type that cannot handle it.

**CORRECT**

overflow

1. /\***XXXXXXXXXXXXX**\*/

**CORRECT**

Block Comment

1. narrowing (of types)

**CORRECT**

casting a data type from a large range to a smaller range - Java does this automatically

1. -names that refer to values or names - letters, digits, \_, and $.  
   -rules for creating a name in a program

**CORRECT**

identifier

1. name of type

**CORRECT**

char

1. Assembler

**CORRECT**

a device used to translate assembly-language programs into machine code

1. operators

**CORRECT**

+, -, \*, /, %

1. Numbers with a decimal point (var double)

**CORRECT**

floating-point number

1. ++

**CORRECT**

increment operator

1. when else matches with the most recent if statement

**CORRECT**

dangling else ambiguity

1. <, <=, ==, !=, >, >=

**THE ANSWER**

Relational Operators (Boolean)

1. \"

**CORRECT**

escape sequence

1. Augmented assignment operators

**CORRECT**

+=, -=, \*\*=, /= and %= (i+= 8 is i = i + 8)

1. expression

**CORRECT**

represents a computation involving values, variables, and operators that, taking them together, evaluates to a value

1. an operation that converts a value of one data type into a value of another data type

**CORRECT**

casting

1. denotes names

**THE ANSWER**

final

1. Reserved words that have a specific meaning in java and cannot be used for variables

**CORRECT**

keyword

1. int

**THE ANSWER**

name of type

1. variable name

**CORRECT**

Bool

1. scope of a variable

**CORRECT**

The part of a program where the variable can be referenced

1. the values operated on by a operator

**CORRECT**

operands

1. determine the order in which operators are evaluated

**CORRECT**

operator associativity

1. -128 to 127

**CORRECT**

byte type

1. Assembly Language

**CORRECT**

uses a short descriptive word to represent each of the machine-language instructions

1. consists of a set of separate programs, each invoked from a command line, for developing and testing Java programs

**CORRECT**

Java Development Toolkit

1. a very large int, more precise

**CORRECT**

long type

1. 3.14159E1

**CORRECT**

floating point/pi

1. imports all the classes in a package by using a \***(import java.util.**\*;)

**CORRECT**

wildcard import

1. boolean operators

**CORRECT**

!, &&, ||, ^

1. Variable

**THE ANSWER**

identifier

1. A library in Java that contains predefined classes and interfaces

**CORRECT**

Application Program Interface ( API)

1. Similar to machine instructions, but can run on any platform with a JVM

**CORRECT**

Bytecode

1. assignment operator

**CORRECT**

=

1. on a program denoted by //xxxxx or /\***xxxx**\*/

**CORRECT**

comment

1. final keyword

**CORRECT**

denotes a value as a constant

1. Constant value directly in a program that stands for itself

**CORRECT**

Literal

1. Boolean Expression

**THE ANSWER**

An expression that evaluates a Boolean value to be true or false

1. nextDouble

**CORRECT**

method that is applied to objects of Scanner

1. constant

**CORRECT**

a number in the program that never changes, denoted by "final"

1. selection statement

**CORRECT**

statements that let you choose actions with alternative choices

1. Byte

**CORRECT**

8 bits to 1 byte

1. predecrement

**CORRECT**

-- placed before variable. decreases variable by one, then uses it in the expression

1. statement terminator

**CORRECT**

;

1. interpreter

**CORRECT**

translates source code into machine code

1. a high-level program's code

**CORRECT**

source code/program

1. widening (of types)

**CORRECT**

casting from a small type to a larger type, this is done manually

1. import statement

**THE ANSWER**

preprocessor

1. fall-through behavior

**CORRECT**

using no breaks in a switch

1. name of a type

**THE ANSWER**

float

1. int type

**CORRECT**

an exact number, 1, 4 or 10

1. String

**CORRECT**

a type

1. util

**CORRECT**

a class name in the system library that contains different java functions

1. Block

**CORRECT**

anything inside of a {xxxxxx}

1. postdecrement

**CORRECT**

-- placed after variable. uses original variable in expression then decreases by 1

1. can be true or false

**CORRECT**

Boolean Value

1. escape character

**CORRECT**

\n

1. ? : for if statement shorthand

**CORRECT**

conditional operator

1. statement

**CORRECT**

instructions for a high-level program

1. IPO

**CORRECT**

input, process, output - describes simple code

1. do, else, and break

**CORRECT**

keywords

1. Occurs when the user inputs a value the program cannot handle

**THE ANSWER**

input error

1. an environment for developing Java programs

**CORRECT**

Integrated development environment

1. keyword

**CORRECT**

abstract is a

1. primitive data type

**CORRECT**

int, real numbers, characters and booleans

1. scientific notation

**CORRECT**

Floating point

1. ++ placed after variable. uses original variable in expression then increases by 1

**CORRECT**

postincrement

1. real numbers, decimal places, twice as precise as float

**CORRECT**

double type

1. decrement operator

**CORRECT**

--

1. assignment statement

**CORRECT**

evaluates to the value to be assigned to a variable (=)

1. illegal identifier

**CORRECT**

4thQtrSales

1. represents a value stored in the computers memory

**CORRECT**

variable

1. a constant value that appears directly in a program

**CORRECT**

literal

1. short circuit operator

**CORRECT**

same as lazy operator - && or || (and, or)

1. Binary digits

**CORRECT**

Bit

1. bytecode verifier

**CORRECT**

checks the validity of a bytecode

1. errors that cause a program to terminate early, an impossible operation is detected

**CORRECT**

runtime error

1. ++ placed before variable. increases variable by one, then uses it in the expression

**CORRECT**

preincrement