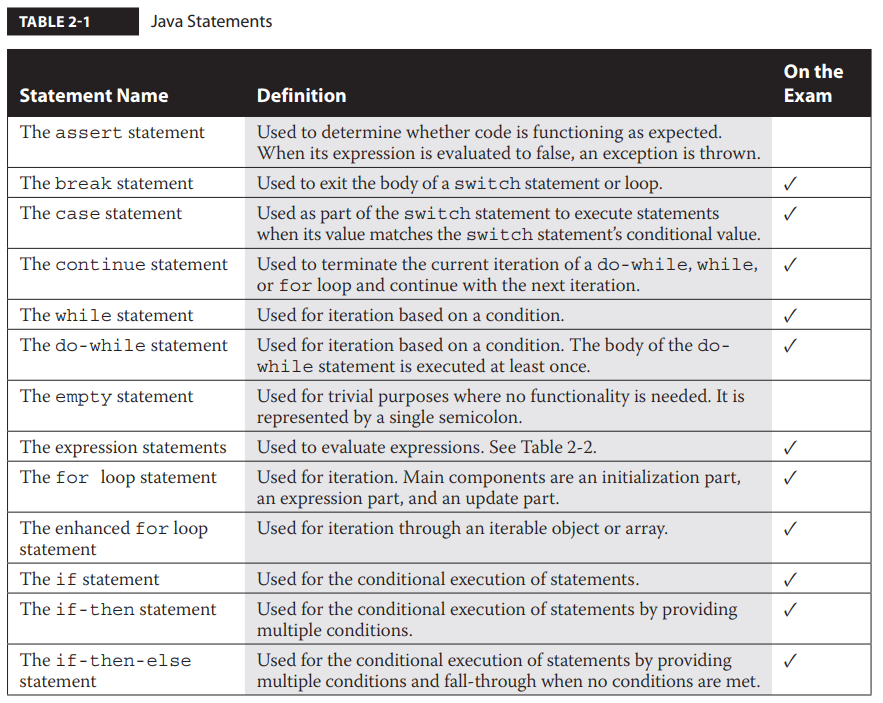
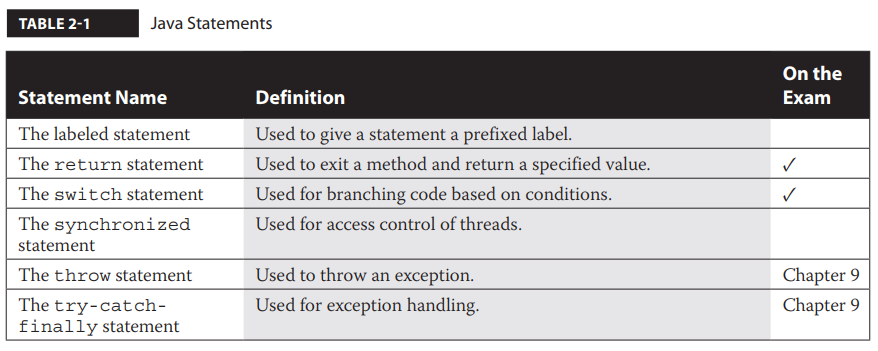
**2**

**Programming with Java Statements**





Expression statements are used for the evaluation of expressions. The assignment expression statements allow assignments to be performed on variables.

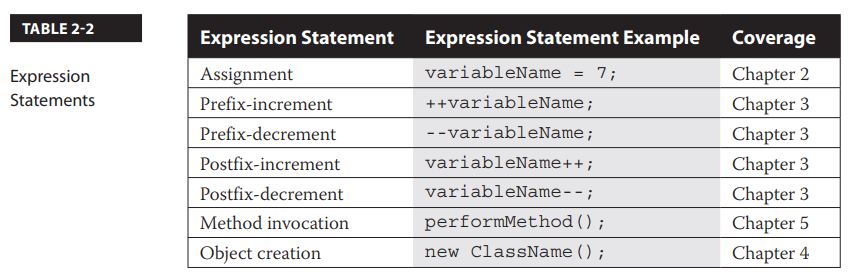
Conditional statements, also known as decision statements, assist in directing the flow of control when a decision needs to be made. Conditional statements include the if, if-then, if-then-else, and switch statements.

Iteration statements provide support in looping through blocks of code. Iteration statements include the for loop, the enhanced for loop, the while statement, and the do-while statement.

Transfer of control statements provide a means of stopping or interrupting the normal flow of control. Transfer of control statements include the continue, break, and return statements. Transfer of control statements are always seen within other types of statements.

**Understand Assignment Statements**

An assignment statement sets a value within a variable. All assignment statements are considered to be expression statements. Expressions in Java are anything that has a value or is reduced to a value. All expressions can be used as statements; the only requirement is that they end with a semicolon.



**The Assignment Expression Statement**

Assignment statements, are designed to assign values to variables. All assignment statements must be terminated with a semicolon.

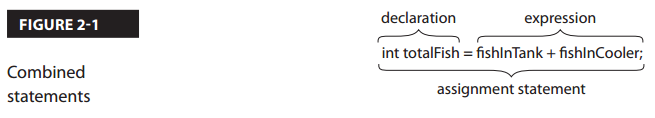
*variable* = *value*;

On the left is the variable that will be associated with the memory and type necessary to store the value. On the right is a literal value. If an expression is on the right, such as (1+2), it must be evaluated down to its literal value before it can be assigned. Lastly, an equal sign resides between the variable and value of an assignment statement.

*int variableName; // Declaration of an integer*

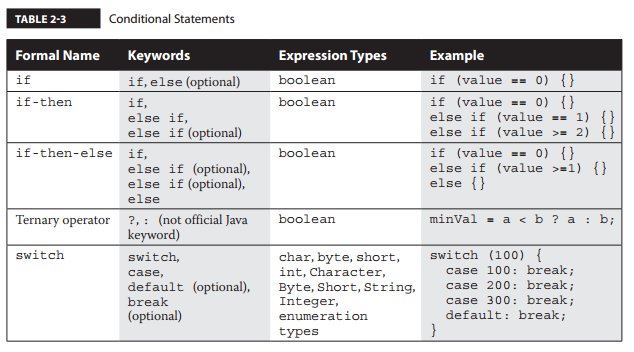
*variableName = 100; // Assignment expression statement*

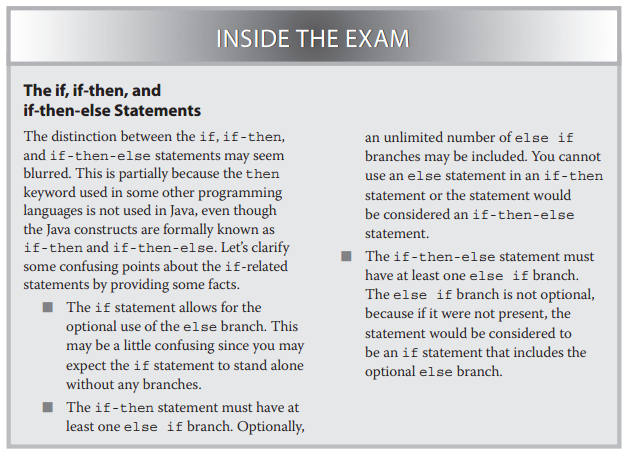
Trying to save an invalid literal to a declared primitive type variable will result in a compiler error. For example, the compilation error Exception in thread "xxxx" java.lang.RuntimeException: Uncompilable source code - incompatible types…



**Create and Use Conditional Statements**

Conditional statements are used when there is a need for determining the direction of flow based on conditions. Conditional statements include the if, if-then, if-then-else, and switch statements.





**The if Conditional Statement**

The if statement is designed to conditionally execute a statement or conditionally decide between a choice of statements. The if statement will execute only one statement upon the condition unless braces are supplied. Braces, also known as curly brackets, allow for multiple enclosed statements to be executed. This group of statements is also known as a *block*. The expression that is evaluated within if statements must evaluate to a boolean value or the application will not compile. The else clause is optional and may be omitted.

if (*expression*)

*statementA*;

else

*statementB*;

Even though relational operators (such as >=) are commonly used, assignment statements are always allowed.

if (bValue = false)

System.out.println("TRUE");

else

System.out.println("FALSE")

The assignment statements of all primitives will return their primitive values:

if (i=1) {}; // will not compile

**The if-then Conditional Statement**

The if-then conditional statement is used when multiple conditions need to flow through a decision-based scenario.

if (*expressionA*)

*statementA*;

else

if (*expressionB*)

*statementB*;

**The if-then-else Conditional Statement**

As with the if and if-then statements, all expressions must evaluate to true or false as the expected primitive type is boolean. The main difference in the if-then-else statement is that the code will fall through to the final stand-alone else when the expression fails to return true for any condition. Each statement may  
optionally be a group of statements enclosed in braces. There is no limit to the number of else if clauses.

if (*expressionA*)

*statementA*;

else if (*expressionB*)

*statementB*;

else if (*expressionC*)

*statementC*;

…

else

*statementZZ;*