Chapter 2:

Java support two different kinds of data: **primitive** data and **objects**

Primitive Types: There are eight primitive data types in Java

- **boolean** (fundamental)
- **byte**
- **char** (fundamental)
- **double** (fundamental)
- **float**
- **int** (fundamental)
- **long**
- **short**

Expression: A simple value or a set of operations that produces a value

Evaluation: The process of obtaining the value of an expression

Arithmetic operator: + addition, − subtraction, * multiplication, / division, % remainder

Operands: The values used in an expression

Literals: The simplest expressions

- **integer literal:** is a sequence of digits with or without a leading sign
- **floating-point literal:** includes a decimal point
- **character literal:** is one character enclosed in single quotation marks
- **boolean literal:** logic deals with just two possibilities: true and false

mod operator (special cases and useful applications):

- **Numerator smaller than denominator:** produces the numerator
- **Numerator of 0:** produces 0
- **Denominator of 0:** throw an ArithmeticException error
- **Number is even or odd:** number % 2 is 0 for evens, number % 2 is 1 for odds
- **Finding individual digits of a number:** number % 10 is the final digit
    - 12345 % 10 = 5
    - 12345 % 100 = 45
    - 12345 % 1000 = 345
    - 12345 % 10000 =2345
    - 12345 % 100000 =12345

Precedence: The binding power of an operator, which determines how to group parts of an expression.

Description	Operators
unary operators	!, ++, —, +, –
multiplicative operators	*, /, %
additive operators	+, –
relational operators	<, >, <=, >=
equality operators	==, !=
logical AND	&&
logical OR	||
assignment operators	=, +=, –=, *=, /=, %=, &&=, ||=

casting : You request a cast by putting the name of the type you want to cast to in parentheses in front of the value you want to cast.

- (int)2.89 = 2
- (double)2 = 2.0

Variable: A memory location with a name and a type that stores a value.

Declaration: A request to set aside a new variable with a given name and type.

Assignment: is the process of giving a value to a variable

Here is an example of declaring multiple variables and assigning some of them values:

- double height = 70, weight = 195, bmi;

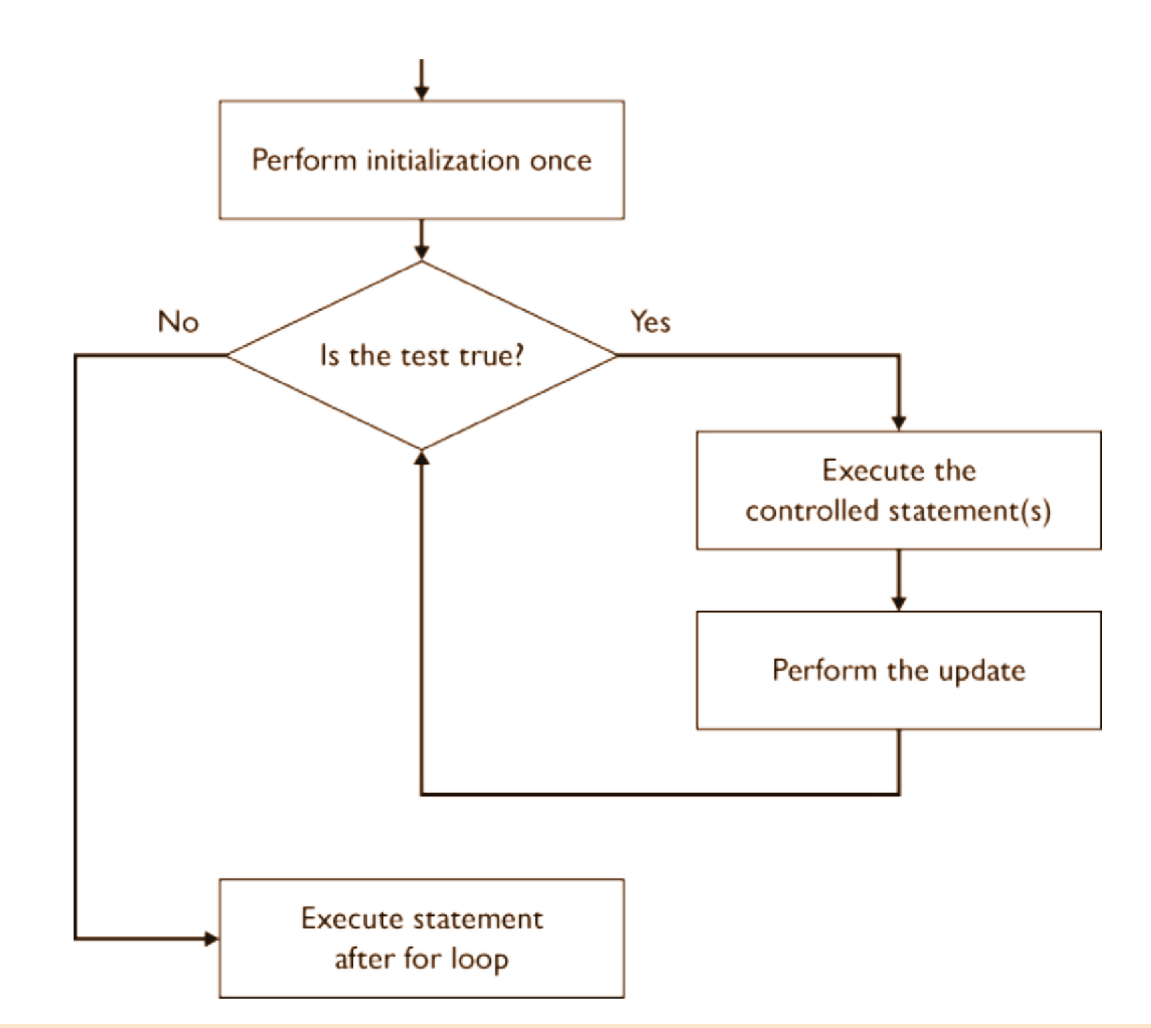
String Concatenation: Combining several strings into a single string, or combining a string with other data into a new, longer string. (using the addition (+) operator)

Useful Shorthand Operators:

Long version	shorthand
x = x + 1;	x += 1;
x = x - 1;	x -= 1;
x = x * 1;	x *= 1;
x = x / 1;	x /= 1;
x = x % 1;	x %= 1;

++/-- pre and post: The pre- and post- variations both have the same overall effect. When you increment or decrement, there are really two values involved: the original value that the variable had before the increment or decrement operation, and the final value that the variable has after the increment or decrement operation. The post- versions evaluate to the original (older) value and the pre- versions evaluate to the final (later) value.

Flow of for loop:



Control Structure: A syntactic structure that controls other statements

Iteration: Each execution of the controlled statement of a loop

Scope: The part of a program in which a particular declaration is valid

- The simple rule is that the scope of a variable declaration extends from the point where it is declared to the right curly brace that encloses it.

Local Variable: A variable declared inside a method that is accessible only in that method

Localizing Variables: Declaring variables in the innermost (most local) scope possible.

- In general, you will want to declare variables in the most local scope possible for security purposes.
- Localizing variables leads to some duplication (and possibly confusion) but provides more security.

Class Constant: A named value that cannot be changed. A class constant can be accessed anywhere in the class

- Constants are declared with the keyword final, which indicates the fact that their values cannot be changed once assigned
- public static final <type> <name> = <expression>;