



COMMON OPERATORS

Frequently Used, Straightforward Behavior



ASSIGNMENT OPERATOR

- NOT the same as = in algebra
- Stores the expression (i.e., the expression) on the right side in the variable on the left side
- Examples:
 - `x = y + 5;`
 - `int a = y + 5;`
 - `int z = x = y + 5;`
 - `w = x = y = z = 0;`



ARITHMETIC OPERATORS

- Generally behave as they do in algebra (i.e., as you would expect of them)
- + Addition
- - Subtraction
- * Multiplication
- / Division
- % Modular (modulo, remainder)

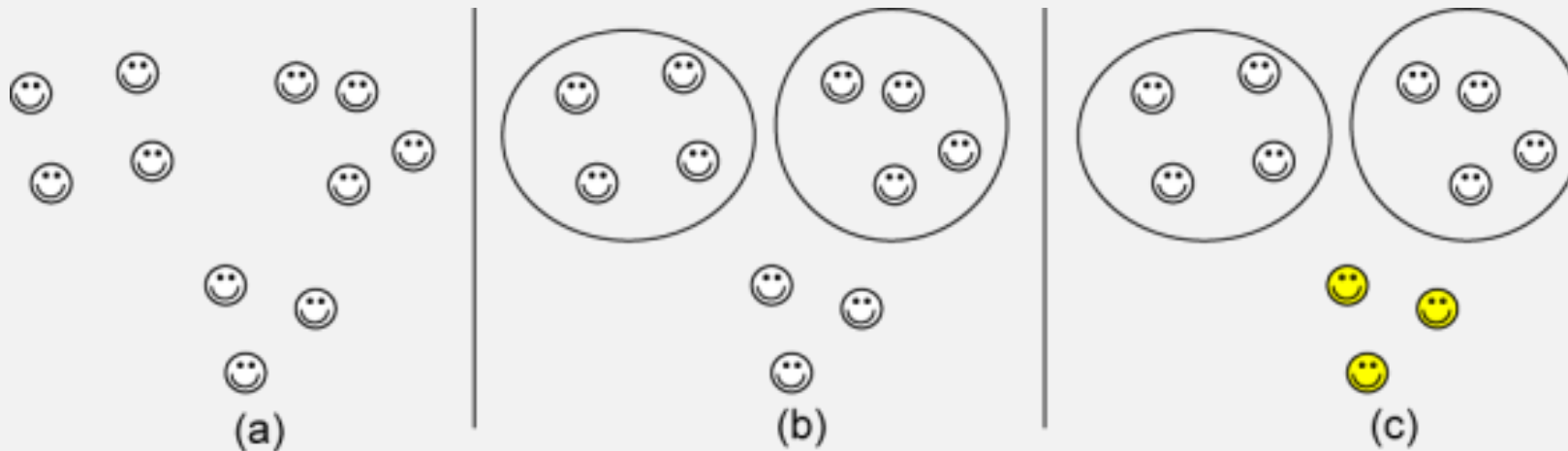


THE DIVISION OPERATOR

- If one or both operands are floating point values (e.g., float or double), the result is a floating point value
 - $3.14 / 2.7$
 - $1.0 / 3$
 - $1 / 3.0$
- *If both operands are integer (char, short, int, or long), the result is a truncated integer*
 - $1 / 3$ is 0
 - $999 / 1000$ is 0

THE MODULAR OPERATOR

- Also known as the remainder operator
- Begin by performing long division but express the results as a quotient and a remainder; discard the quotient; the result is the remainder: $11 \% 4 = 3$





CASTING OPERATOR

- The compiler will automatically perform some type conversions, called a *type promotion*:
 - `double max = 95;`
 - `double x = sqrt(2);`
 - Explicit cast:
 - `int score = (int)95.5;`
 - `int score = int(95.5);`
 - `(double)score / 10`
 - `double(score) / 10`
 - `(int)(3.14 + 2.7)`
- `double(score / 10) ????`
- `(int)3.14 + 2.7 ????`



LIMITS OF CASTING

- Casting an int to a double is okay
- Casting a double to an int is okay
- What does it mean to
 - Cast an int to a string
 - Cast a string to an int
 - Person to a Square
 - A Square to a Person

- Informal Casting Rule:

To cast from one data type to another, the two data types, the new and the current types, must be "sort of the same" to begin with.