Expressions with Common Operators

Expressions

As we mentioned previously, an expression is a combination of values and operations which evaluates to a value. We have already seen the simplest expressions—numerical constants, which evaluate to themselves. We can also use mathematical operators, such as +, –, *, and / to carry out arithmetic operations. For example, 7 + 3 evaluates to 10 and 4 * 6 + 9 * 3 evaluates to 51. These operators have the standard rules of precedence—multiplication and division occur before addition and subtraction—and associativity: 4 – 3 – 1 means (4 – 3) – 1 not 4 – (3 –1). Parenthesis may be used to enforce a specific order of operations—4 * (6 + 9) * 3 evaluates to 180.

Another common operator which you may not be as familiar with is the modulus operator, %. The modulus operator evaluates to the remainder when dividing the first operand by the second. That is a % b (read “a modulus b”, or “a mod b” for short) is the remainder when a is divided by b. For example, 19 % 5 = 4 because 19 / 5 = 3 with a remainder of 4—3 * 5 = 15, and 19 – 15 = 4.

One slightly tricky thing about division with integers is that dividing an integer by an integer gives an integer. This means that 5 / 2 is 2. Note that we are using *floor* division (ie: we round down the result). This happens because integer can only hold whole numbers. In a later lesson you will learn about other types that can hold numbers like 2.5, so that you can compute with fractional numbers if you need to.

Variables may also appear in expressions. When a variable appears in an expression, it is evaluated to a value by reading the current value out of its box. It is important to note that assignment statements involving variables on the right side are not algebraic equations to be solved—we cannot write x – y = z * q. Note that here, the left side of this statement does not “name a box”. If you want to solve an algebraic equation, you must do so in a step-by-step fashion.

We can, however, write perfectly meaningful assignment statements which are not valid in algebra. For example, a statement such as x = x + 1; is quite common in programming, but has no solution if you think of it as an algebraic equation. In programming, this statement means to take current value of x, add 1 to it, and update x’s value to whatever that result is.

The next video shows the execution of some assignment statements with more complex expressions on their right-hand sides than in previous examples.