Expressions and Statements

Expression

An **expression** is a combination of values, variables, and operators.

```
>>> 5 + 5 # addition expression
10
>>> n = 17 # assignment statement (with a numeric literal 17 as an expression)
>>> n # variable expression
17
>>> n > 10 # boolean expression
True
```

When you execute an expression on Python interpreter prompt, the interpreter **evaluates** it. Evaluating an expression means to find the value of the expression.

Statement

A **statement** is a unit of code that has an effect, like creating a variable or displaying a value.

An **expression** is a **part** of a **statement**. The expression gets evaluated first, and then the statement gets executed.

```
 1  # test.py
 2
 3  n = 17 # assignment statement (with a numeric literal 17 as an expression)
 4  print n # print statement (with variable n as an expression)
 5  print n == 3 # print statement (with n == 3 as an expression)
 6
 7  # ======================
 8  # Output
 9  # ======================
10  17
11  False
```

- Line #3 is an assignment statement where numeric literal `17` is an expression (which is a part of the statement).
- Line #4 is a print statement where variable `n` is an expression (which is a part of the statement).
- Line #5 is a print statement where `n == 3` is an expression (which is a part of the statement).

Main Takeaways

> An **expression** is a combination of values, variables, and operators (e.g. +, -, /, *, =, ==) that Python **evaluates**

> A **statement** is a unit of code that has an effect or carries out certain actions which Python **executes**.