Like all object-oriented languages, Python utilizes variables to store and overwrite information in memory. Traditional variables come in a variety of datatypes, most of which are supported by the standard Python library. Some of these datatypes include, integers, floats, and strings. In this tutorial, we will review these primitive datatypes, as well as cover some examples of how they can be used.

We will first look at one of the two types of numbers: Integers. Variables initialized as an integer datatype can store whole number values, and can also be used to perform arithmetic operations on. To declare an integer variable, we simply have to give our variable a name and assign an integer value to it. Since the name of our variable is up to our choosing, let us give it a name to help suggest what type of value is being stored in it.

myInt = 5

Some other examples can be written like so:

X = 0

I = 3

Integer1 = 10

newInt = 4

When these lines are executed, each of these numbers will be stored in memory under each respective variable name. If we wish to access these numbers later on in our script, we can simply do that using the variables.

Here’s another example:

myVal = 5+3

This will store the value 8 inside of variable myVal. The mathematical operation of the plus sign is resolved before the actual value is assigned. Similary:

myVal = myInt + Integer1

This will store the value 15 in myVal, as we are taking the value 5 in myInt, and adding it to the value 10 in Integer1. It should also be noted that this line will overwrite the previous value of myVal (8) since a new value is being written to this variable (or address) in memory.

The other type of numeric variable, floating point numbers, work very similarly to integers. The only difference is that float variables can store decimal values.

Float1 = 4.5

Float2 = 3.0

Float3 = 3.6 + 5

myFloat = 4.3 + 5.6 – 2

Performing mathematical operations on integers and at least one float will almost always result in an answer of type float.

You may also use the int() and float() functions to convert a number to an integer or float respectively. Note that using the int() function on a float value will not round the value, but instead of remove all numbers after the decimal points.

Int1 = int(5.7)

Print(int1)

Float1 = float(3)

Print(float1)

Finally, we take a look at strings. Strings are essentially a collection of characters. They can be used to store words as a sequence of characters. Strings are defined with either single quotes or double quotes, although both can be used interchangeably. If a string contains single quotes within it however, it might be a good idea to just use double quotes.

Some example:

myString = ‘hello’

newString = “Hi my name is”

s1 = “Don’t fear the reaper”

s2 = “12345”

Simple operators can be used on strings:

S3 = “Don’t fear the reaper” + “ “ + “12345”

Print(s3)

However, you can’t perform operations across integers/floats and strings. So the following is not allowed:

S4 = “Hello” + 2

One final tip is that multiple variable assignments can be done simultaneously on the same line, like so:

Int1 = 4, float1 = 2.3, string1 = “Hello”

Print(int1)

Print(float1)

Print(string1)

To recap, there are three main primitive datatypes used in Python: Integers, floating point numbers, and strings. Integers are used to store whole numbers, while floats are used to store decimal values. Strings contain a collection of characters. Mathematical operations between integers and floats will usually result in floating point values. The int() and float() functions can be used to convert numeric values to either an integer or a floating point value respectively. Mathematical operations cannot be performed between integers/floats and strings.