

# Numbers and more in Python!

In this lecture, we will learn about numbers in Python and how to use them.

We'll learn about the following topics:

- 1.) Types of Numbers in Python
- 2.) Basic Arithmetic
- 3.) Differences between classic division and floor division
- 4.) Object Assignment in Python

## Types of numbers

Python has various "types" of numbers (numeric literals). We'll mainly focus on integers and floating point numbers.

Integers are just whole numbers, positive or negative. For example: 2 and -2 are examples of integers.

Floating point numbers in Python are notable because they have a decimal point in them, or use an exponential (e) to define the number. For example 2.0 and -2.1 are examples of floating point numbers. 4E2 (4 times 10 to the power of 2) is also an example of a floating point number in Python.

Throughout this course we will be mainly working with integers or simple float number types.

Here is a table of the two main types we will spend most of our time working with some examples:

Examples	Number "Type"
1,2,-5,1000	Integers
1.2,-0.5,2e2,3E2	Floating-point numbers

Now let's start with some basic arithmetic.

## Basic Arithmetic

In [1]:

```
# Addition  
2+1
```

Out[1]:

3

In [2]:

```
# Subtraction  
2-1
```

Out[2]:

1

In [3]:

```
# Multiplication  
2*2
```

Out[3]:

4

In [4]:

```
# Division  
3/2
```

Out[4]:

1.5

In [5]:

```
# Floor Division  
7//4
```

Out[5]:

1

**Whoa! What just happened? Last time I checked, 7 divided by 4 equals 1.75 not 1!**

The reason we get this result is because we are using "floor" division. The // operator (two forward slashes) truncates the decimal without rounding, and returns an integer result.

**So what if we just want the remainder after division?**

In [6]:

```
# Modulo  
7%4
```

Out[6]:

3

4 goes into 7 once, with a remainder of 3. The % operator returns the remainder after division.

**Arithmetic continued**

In [7]:

```
# Powers  
2**3
```

Out[7]:

8

In [8]:

```
# Can also do roots this way  
4**0.5
```

Out[8]:

2.0

In [9]:

```
# Order of Operations followed in Python  
2 + 10 * 10 + 3
```

Out[9]:

105

In [10]:

```
# Can use parentheses to specify orders  
(2+10) * (10+3)
```

Out[10]:

156

## Variable Assignments

Now that we've seen how to use numbers in Python as a calculator let's see how we can assign names and create variables.

We use a single equals sign to assign labels to variables. Let's see a few examples of how we can do this.

In [11]:

```
# Let's create an object called "a" and assign it the number 5  
a = 5
```

Now if I call a in my Python script, Python will treat it as the number 5.

In [12]:

```
# Adding the objects  
a+a
```

Out[12]:

10

What happens on reassignment? Will Python let us write it over?

In [13]:

```
# Reassignment  
a = 10
```

In [14]:

```
# Check  
a
```

Out[14]:

10

Yes! Python allows you to write over assigned variable names. We can also use the variables themselves when doing the reassignment. Here is an example of what I mean:

In [15]:

```
# Check  
a
```

Out[15]:

10

In [16]:

```
# Use A to redefine A  
a = a + a
```

In [17]:

```
# Check  
a
```

Out[17]:

20

The names you use when creating these labels need to follow a few rules:

1. Names can not start with a number.
2. There can be no spaces in the name, use `_` instead.
3. Can't use any of these symbols : `'",<>/?|\()!@#$$%^&*~--+`
4. It's considered best practice (PEP8) that names are lowercase.
5. Avoid using the characters `'l'` (lowercase letter el), `'O'` (uppercase letter oh), or `'I'` (uppercase letter eye) as single character variable names.
6. Avoid using words that have special meaning in Python like `"list"` and `"str"`

Using variable names can be a very useful way to keep track of different variables in Python. For example:

In [18]:

```
# Use object names to keep better track of what's going on in your code!
my_income = 100

tax_rate = 0.1

my_taxes = my_income*tax_rate
```

In [19]:

```
# Show my taxes!
my_taxes
```

Out[19]:

10.0

So what have we learned? We learned some of the basics of numbers in Python. We also learned how to do arithmetic and use Python as a basic calculator. We then wrapped it up with learning about Variable Assignment in Python.

Up next we'll learn about Strings!