Python course materials

# Variable Assignment

## Rules for variable names

* names can not start with a number
* names can not contain spaces, use \_ intead
* names can not contain any of these symbols:
* :'",<>/?|\!@#%^&\*~-+
* it’s considered best practice ([PEP8](https://www.python.org/dev/peps/pep-0008/#function-and-variable-names)) that names are lowercase with underscores
* avoid using Python built-in keywords like list and str
* avoid using the single characters l (lowercase letter el), O (uppercase letter oh) and I (uppercase letter eye) as they can be confused with 1 and 0

## Dynamic Typing

Python uses *dynamic typing*, meaning you can reassign variables to different data types. This makes Python very flexible in assigning data types; it differs from other languages that are *statically typed*.

my\_dogs = 2

my\_dogs

2

my\_dogs = ['Sammy', 'Frankie']

my\_dogs

['Sammy', 'Frankie']

### Pros and Cons of Dynamic Typing

#### Pros of Dynamic Typing

* very easy to work with
* faster development time

#### Cons of Dynamic Typing

* may result in unexpected bugs!
* you need to be aware of type()

## Assigning Variables

Variable assignment follows name = object, where a single equals sign = is an *assignment operator*

a = 5

a

5

Here we assigned the integer object 5 to the variable name a.Let’s assign a to something else:

a = 10

a

10

You can now use a in place of the number 10:

a + a

20

## Reassigning Variables

Python lets you reassign variables with a reference to the same object.

a = a + 10

a

20

There’s actually a shortcut for this. Python lets you add, subtract, multiply and divide numbers with reassignment using +=, -=, \*=, and /=.

a += 10

a

30

a \*= 2

a

60

## Determining variable type with type()

You can check what type of object is assigned to a variable using Python’s built-in type() function. Common data types include: \* **int** (for integer) \* **float** \* **str** (for string) \* **list** \* **tuple** \* **dict** (for dictionary) \* **set** \* **bool** (for Boolean True/False)

type(a)

int

a = (1,2)

type(a)

tuple

## Simple Exercise

This shows how variables make calculations more readable and easier to follow.

my\_income = 100  
tax\_rate = 0.1  
my\_taxes = my\_income \* tax\_rate

my\_taxes

10.0

Great! You should now understand the basics of variable assignment and reassignment in Python.Up next, we’ll learn about strings!