Python Syntax

- ✓ The term syntax is referred to a set of rules and principles that describes the structure of a language
- ✓ **PEP 8** (Python Enhancement Proposal) defines all the set of rules that are used to create sentences in Python programming
- ✓ You can find all the PEP Index at : https://www.python.org/dev/peps/

Various structures in Python syntax :

1. Python Line Structure

A Python program is divided into a number of logical lines and every logical line is terminated by the NEWLINE

2. Python Multiline Statements (Line Joining)

Sometimes, you may want to split a statement over two or more lines. It may be to aid readability.

You can do so in the following ways :

– Use a backward slash (\)

– Put the String in Triple Quotes (“‘ ”’)

3. Python Comments

Comments are used to :

- ✓ Explain Python code.
- ✓ Make the code more readable.
- ✓ To prevent execution when testing code.

Single Line Comments :

Comments starts with a `#`, and Python will ignore them

Multi Line Comments :

- ✓ Python does not really have a syntax for multi line comments.
- ✓ To add a multiline comment you could insert a `#` for each line
- ✓ Since Python will ignore string literals that are not assigned to a variable, you can add a multiline string (triple quotes) in your code, and place your comment inside it

4. Python Docstrings

Python docstrings are the string literals that appear right after the definition of a function, method, class, or module

Python __doc__ attribute :

- ✓ Whenever string literals are present just after the definition of a function, module, class or a method, they are associated with the object as their **__doc__** attribute. We can later use this attribute to retrieve this docstring.
- ✓ We can also use the **help()** function to read the docstrings associated with various objects.

5. Python Indentation

- ✓ Indentation refers to the spaces at the beginning of a code line.
- ✓ Since Python doesn't use curly braces to delimit blocks of code, this Python Syntax is mandatory
- ✓ You can indent code under a function, loop, or class
- ✓ Python will give you an error if you skip the indentation
- ✓ You have to use the same number of spaces in the same block of code, otherwise Python will give you an error

6. Python Multiple Statements in One Line

You can also fit in more than one statement on one line. Do this by separating them with a semicolon. But you'd only want to do so if it supplements readability

7. Python Quotations

Python supports the single quote and the double quote for string literals. But if you begin a string with a single quote, you must end it with a single quote. The same goes for double-quotes.

8. Python Blank Lines

If you leave a line with just white space , the interpreter will ignore it

9. Python Variables

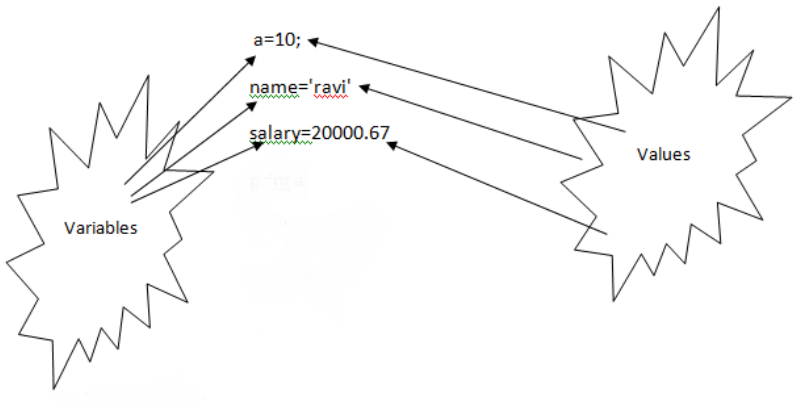
- ✓ Variable is a name which is used to **refer memory location**.
- ✓ A variable is a container for a value. It can be assigned a name, you can use it to refer to it later in the program.
- ✓ In Python, we don't need to specify the type of variable because Python is a **dynamically typed** language . Based on the value assigned, the interpreter decides its data type

Python Variables Naming Rules :

- ✓ The first character of the variable must be an alphabet or underscore (_).
- ✓ All the characters except the first character may be an alphabet of lower-case(a-z), upper-case (A-Z), underscore or digit (0-9).
- ✓ Variable name must not contain any white-space, or special character (!, @, #, %, ^, &, *).
- ✓ Variable name must not be similar to any keyword defined in the language.
- ✓ Python is case-sensitive, So Python Variable names are also case sensitive for example name, and Name is not the same.
- ✓ It is recommend to use lowercase letters for variable name.
- ✓ Examples of valid identifiers : a123, _n, n_9, etc.
- ✓ Examples of invalid identifiers : 1a, n%4, n 9, etc.

Assigning and Reassigning Python Variables :

To assign a value to Python variables, you don't need to declare its type. You name it according to the rules stated and type the value after the **equal sign(=)**



Multiple Assignment :

- ✓ Python allows us to assign a value to multiple variables in a single statement which is also known as multiple assignment.
- ✓ We can apply multiple assignments in two ways either by assigning a single value to multiple variables or assigning multiple values to multiple variables.

I. Assigning single value to multiple variables

Eg: x=y=z=50

II. Assigning multiple values to multiple variables:

Eg: a,b,c=5,10,15

The values will be assigned in the order in which variables appears.

Deleting Variables :

You can also delete Python variables using the keyword **'del'**

10. Python Identifiers :

- ✓ Identifier is a name given to the fundamental building blocks (a function, a class, a module, or any other object) in a program
- ✓ An identifier can be as long as you want. According to the docs, you can have an identifier of infinite length. However, the PEP-8 standard sets a rule that you should limit all lines to a maximum of 79 character

Best Practices in Identifiers in Python :

While it's mandatory to follow the rules, it is also good to follow some recommended practices:

- ✓ Begin class names with an uppercase letter, begin all other identifiers with a lowercase letter
- ✓ Begin private identifiers with an underscore (_); Note that this doesn't make a variable private, but discourages the user from attempting to access it
- ✓ Put __ around names of magic methods (use leading and trailing double underscores), avoid doing this to anything else. Also, built-in types already use this notation.
- ✓ Use leading double underscores only when dealing with private variable

11. **Python Keywords**

- ✓ Python has a set of keywords that are reserved words that cannot be used as variable names, function names, or any other identifiers
- ✓ In Python, keywords are case sensitive.
- ✓ There are 33 keywords in Python 3.7. This number can vary slightly over the course of time.
- ✓ All the keywords except True, False and None are in lowercase and they must be written as they are.

The list of all the keywords is given below.

Keyword	Description
and	A logical operator
as	To create an alias
assert	For debugging
break	To break out of a loop
class	To define a class
continue	To continue to the next iteration of a loop
def	To define a function
del	To delete an object
elif	Used in conditional statements, same as else if
else	Used in conditional statements
except	Used with exceptions, what to do when an exception occurs
False	Boolean value, result of comparison operations
finally	Used with exceptions
for	To create a for loop
from	To import specific parts of a module
global	To declare a global variable
if	To make a conditional statement
import	To import a module
in	To check if a value is present in a list, tuple, etc.
is	To test if two variables are equal

lambda	To create an anonymous function
None	Represents a null value
nonlocal	To declare a non-local variable
not	A logical operator
or	A logical operator
pass	A null statement, a statement that will do nothing
raise	To raise an exception
return	To exit a function and return a value
True	Boolean value, result of comparison operations
try	To make a try...except statement
while	To create a while loop
with	Used to simplify exception handling
yield	To end a function, returns a generator

keyword.iskeyword(s) :

Return true if s is a Python keyword.

keyword.kwlist :

Gives Sequence containing all the keywords defined for the interpreter.

The joy of coding Python should be in seeing

short, concise, readable classes that express a lot of action in a small amount

of clear code — not in reams of trivial code that bores the reader to death

- Guido van Rossum