What is Python?

Python is a popular programming language. It was created by **Guido van Rossum**, and released in 1991.

It is used for:

- web development (server-side),
- software development,
- mathematics,
- system scripting.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- Python works on different **platforms** (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax **similar to the English** language.
- Python has syntax that allows developers to write programs with **fewer lines** than some other programming languages.
- Python runs on an **interpreter** system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a **procedura**l way, an **object-oriented** way or a **functional** way.Good to know
- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
- In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment **(IDE)**, such as **Thonny**, **Pycharm**, **Netbeans** or **Eclipse** which are particularly useful when managing larger collections of Python files.Python Syntax compared to other programming languages
- Python was designed for readability, and has some similarities to the English language with influence from mathematics.

- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on **indentation**, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Comments in Python:

-python commets satrtedwitha #

```
#This is a comment
print("Hello, World!")
```

Creating Variables

Python has no command for declaring a variable.
A variable is created the moment you first assign a value to it.

```
x = 5
y = "John"
print(x)
print(y)
```

```
a=10
b=20
c=a+b
print (c)
```

Casting

If you want to specify the data type of a variable, this can be done with casting.
Example

```
x = str(3)    # x will be '3'
y = int(3)    # y will be 3
z = float(3)  # z will be 3.0
a=int(input("Enter the value : "))
```

Get the Type

You can get the data type of a variable with the **type()** function.

```
type(x)  # it will return str
```

Case Sensitivity: the variables are case sensitive

```
a=10
A=20
print (a)
print(A)
```

Variable Names

A variable can have a short name (like x and y) or a more descriptive name (age, carname, total_volume).
Rules for Python variables:

- A variable name must start with a letter(a) or the underscore(_a) character
- A variable name cannot start with a number
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and _)
- Variable names are case-sensitive (age, Age and AGE are three different variables)

Multi Words Variable Names
Variable names with more than one word can be difficult to read.

There are several techniques you can use to make them more readable:
-Camel Case
Each word, except the first, starts with a capital letter:
myVariableName = "John"
-Pascal Case
Each word starts with a capital letter:
MyVariableName = "John"
-Snake Case
Each word is separated by an underscore character:
my_variable_name = "John"

Many Values to Multiple Variables
Python allows you to assign values to multiple variables in one line:
Example
```
x, y, z = "Orange", "Banana", "Cherry"
print(x)
print(y)
print(z)
```

One value to Multiple Variable:
```
x=y=z="Orange"
print(x)
print(y)
print(z)
```

Unpack a Collection
If you have a collection of values in a list, tuple etc. Python allows you extract the values into variables. This is called *unpacking*.

Example
Unpack a list:

```
fruits = ["apple", "banana", "cherry"]
x, y, z = fruits
print(x)
print(y)
print(z)
```