PYTHON

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.

Terms in Python

- **INDENTION** - refers to the spaces at the beginning of a code line; to indicate a block of code.
- **VARIABLES** are created when you assign a value to it.
- **COMMENTS** - in-code documentation; Comments start with a "#", and Python will render the rest of the line as a comment.

CODE

print() - function to display text or output values.
- *EX: print("Hello World!")*

If you want to print multiple words on the same line, you can use the "end=" parameter:
- *print("Hello World!", end=" ")*
  print("I will print on the same line.")

CASTING

int() - constructs an integer number from an integer literal, a float literal (by removing all decimals), or a string literal (providing the string represents a whole number)

- *x = int(1)   # x will be 1*
- *y = int(2.8)   # y will be 2*
- *z = int("3")   # z will be 3*

float() - constructs a float number from an integer literal, a float literal or a string literal (providing the string represents a float or an integer)

- *x = float(1)   # x will be 1.0*
- *y = float(2.8)   # y will be 2.8*
- *z = float("3")   # z will be 3.0*
- *w = float("4.2") # w will be 4.2*

str() - constructs a string from a wide variety of data types, including strings, integer literals and float literals

- *x = str("s1")   # x will be 's1'*
- *y = str(2)   # y will be '2'*
- *z = str(3.0)   # z will be '3.0'*

DATA TYPES

Text Type:	str
Numeric Types:	int, float, complex
Sequence Types:	list, tuple, range
Mapping Type:	dict
Set Types:	set, frozenset
Boolean Type:	bool
Binary Types:	bytes, bytearray, memoryview
None Type:	NoneType

Example	Data Type
x = "Hello World"	str
x = 20	int
x = 20.5	float
x = 1j	complex
x = ["apple", "banana", "cherry"]	list
x = ("apple", "banana", "cherry")	tuple
x = range(6)	range
x = {"name" : "John", "age" : 36}	dict
x = {"apple", "banana", "cherry"}	set
x = frozenset({"apple", "banana", "cherry"})	frozenset
x = True	bool
x = b"Hello"	bytes
x = bytearray(5)	bytearray
x = memoryview(bytes(5))	memoryview
x = None	NoneType

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 or the underscore 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)
- A variable name cannot be any of the Python keywords.

CONDITION and IF ELSE STATEMENTS

Python supports the usual logical conditions from mathematics:

- Equals: `a == b`
- Not Equals: `a != b`
- Less than: `a < b`
- Less than or equal to: `a <= b`
- Greater than: `a > b`
- Greater than or equal to: `a >= b`

These conditions can be used in several ways, most commonly in "if statements" and loops.

The `elif` keyword is Python's way of saying "if the previous conditions were not true, then try this condition".

DICTIONARIES

- are used to store data values in key:value pairs.
- is a collection which is ordered, changeable and does not allow duplicates.

TUPLE

- A collection similar to a Python list. The primary difference is that we cannot modify a tuple once it is created.

SETS

- A collection of unique data, meaning that elements within a set cannot be duplicated.

Parameters are variables defined in a function declaration. This acts as placeholders for the values (arguments) that will be passed to the function.

Arguments are the actual values that you pass to the function when you call it. These values replace the parameters defined in the function.