1

WHAT IS A VARIABLE IN PYTHON?

🎯 Real-Life Example: Coffee Shop Order

Let's say you're at a coffee shop. You place an order:

```python
customer_name = "Samruddhi"
drink = "Cappuccino"
size = "Medium"
sugar_spoons = 2
price = 180.50
```

What's Happening Here in Python?

- customer_name = "Samruddhi" → "customer_name" is the variable, and "Samruddhi" is its value — it stores the name of the customer.
- drink = "Cappuccino" → "drink" is the variable, and "Cappuccino" is its value — it stores the type of drink ordered.
- size = "Medium" → "size" is the variable, and "Medium" is its value — it stores the size of the drink.
- sugar_spoons = 2 → "sugar_spoons" is the variable, and 2 is its value — it stores the number of sugar spoons.

A variable is like a container
or a label that stores data —
just like how you store your contact number with your friend's name in your phone.

samruddhikamble015@gmail.com

2

IN PYTHON, A VARIABLE IS JUST A NAME THAT REFERS TO AN OBJECT

```
a = [1, 2, 3]          # Point "a" at a new list, [1, 2, 3, 4]
b = a                  # Point "b" at what "a" is pointing to
b += [4]               # Extend the list pointed by "a" and "b" by adding "4" to it
a                      # Returns "[1, 2, 3, 4]"
b                      # Also returns "[1, 2, 3, 4]"
```

is checks if two variables refer to the same object, while == checks if the objects that the variables point to have the same values:

```
a = [1, 2, 3, 4]                    # Point "a" at a new list, [1, 2, 3, 4]
b = a                               # Point "b" at what a is pointing to
b is a                              # Returns True, "a" and "b" refer to the *same* object
b == a              # Returns True, the objects that "a" and "b" are pointing to are *equal*
b = [1, 2, 3, 4]                    # Point b at a new list, [1, 2, 3, 4]
b is a                  # Returns False, "a" and "b" do not refer to the *same* object
b == a              # Returns True, the objects that "a" and "b" are pointing to are *equal*
```

samruddhikamble015@gmail.com

3

IN PYTHON, EVERYTHING IS AN OBJECT.

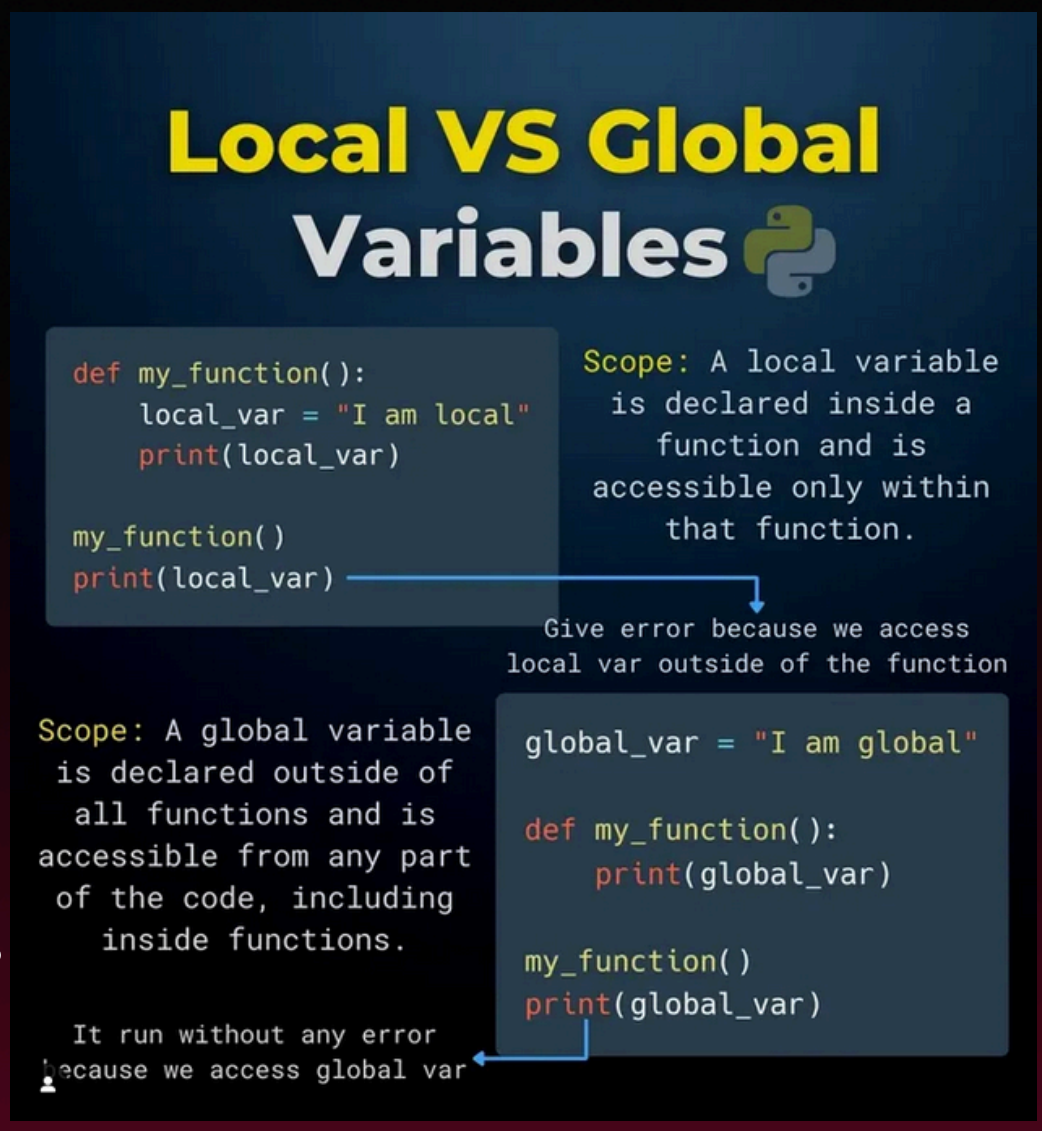
THIS MEANS THAT EVEN NONE (WHICH IS USED TO DENOTE THAT THE VARIABLE DOESN'T POINT TO AN OBJECT YET) IS ALSO AN OBJECT!

```
# DON'T USE THE EQUALITY "==" SYMBOL TO COMPARE OBJECTS TO
NONE

# USE "IS" INSTEAD. THIS CHECKS FOR EQUALITY OF OBJECT IDENTITY.

"ETC" IS NONE                    # RETURNS FALSE

NONE IS NONE                     # RETURNS TRUE
```

samruddhikamble015@gmail.com

4

PYTHON HAS LOCAL AND GLOBAL VARIABLES.



HERE'S AN EXAMPLE OF LOCAL VS. GLOBAL VARIABLE SCOPE:

```
x = 5

def set_x(num):   # Local var x not the same as global variable x
x = num           # Returns 43
x                  # Returns 43

def set_global_x(num):
global x
x                  # Returns 5
x = num           # global var x is now set to 6
x                  # Returns 6
set_x(43)
set_global_x(6)
```

samruddhikamble015@gmail.com