📘 Lecture 10: Local vs. Global Variables in Python

🧠 What Are Variables?

Variables store **data** in Python.
They can live either:

- Inside a function (**Local**)
- Outside a function (**Global**)

🔹 1. Local Variables

- Declared **inside** a function.
- Can only be used **within that function**.
- Think of a **classroom whiteboard** — only visible to students in that room.

```
python
CopyEdit
def my_function():
    y = 5  # Local
    print("Inside function: y =", y)

my_function()

# print(y)  ✖ Error: y is not defined outside
```

🔹 2. Global Variables

- Declared **outside** all functions.
- Can be used **anywhere** in the program.
- Like a **school notice board** — visible to all.

```
python
CopyEdit
x = 10  # Global

def my_function():
    print("Inside function: x =", x)

my_function()
print("Outside function: x =", x)
```

□ 3. Local vs Global – Combined Example

```
python
CopyEdit
x = 10  # Global

def my_function():
    y = 5  # Local
    print("Inside: x =", x, "y =", y)

my_function()
print("Outside: x =", x)

# print(y) ✕ Error: y is local only
```

□ 4. Understanding Scope

Variable Type	Where Defined	Where Accessible
Local	Inside Function	Only in that function
Global	Outside Function	Anywhere in code

🔄 5. Nested Functions & Enclosing Variables

```
python
CopyEdit
x = 10

def outer():
    y = 5  # Enclosing
    def inner():
        z = 3  # Local
        print("x =", x, "y =", y, "z =", z)
    inner()

outer()
```

□ `x` is global
◍ `y` is enclosing
□ `z` is local to inner()

🔗 6. Modifying Global Variables

Use `global` keyword to modify a global variable inside a function:

```
python
CopyEdit
counter = 0

def increment():
    global counter
    counter += 1

increment()
print("Counter:", counter)  # Output: 1
```

Without `global`, it treats `counter` as a new local variable.

🆚 7. Comparing Variables: `==` vs `is`

```
python
CopyEdit
a = [1, 2, 3]
b = [1, 2, 3]

print(a == b)  # ✅ True (values are equal)
print(a is b)  # ❌ False (different memory)

c = a
print(a is c)  # ✅ True (same memory)
```

Operator	Meaning
`==`	Compare values
`is`	Compare memory locations

📌 Summary

Feature	Local Var	Global Var
Where Defined	In function	Outside all functions
Scope	Inside that function only	Everywhere

Feature	Local Var	Global Var
Accessible in Function?	✅	✅
Modify Inside Function?	✅	Only with `global` keyword

🧠 Final Thoughts

- ✅ Understand **where** and **how** variables are declared
- ✅ Know when to use `global`
- ✅ Avoid global overuse — can cause confusion in large code
- ✅ Practice nested + scope-based examples
