**Namespaces in Python:**

At any given moment, there are at least three nested scopes.

* Scope of the current function which has local names
* Scope of the module which has global names
* Outermost scope which has built-in names

When a reference is made inside a function, the name is searched in the local namespace, then in the global namespace and finally in the built-in namespace.

If there is a function inside another function, a new scope is nested inside the local scope.

def outer\_function():

a = 20

def inner\_function():

a = 30

print('a =',a)

inner\_function()

print('a =',a)

a = 10

outer\_function()

print('a =',a)

**Output:**

a = 30

a = 20

a = 10

def outer\_function():

**global a**

a = 20

def inner\_function():

**global a**

a = 30

print('a =',a)

inner\_function()

print('a =',a)

a = 10

outer\_function()

print('a =',a)

**Output:**

a = 30

a = 30

a = 30