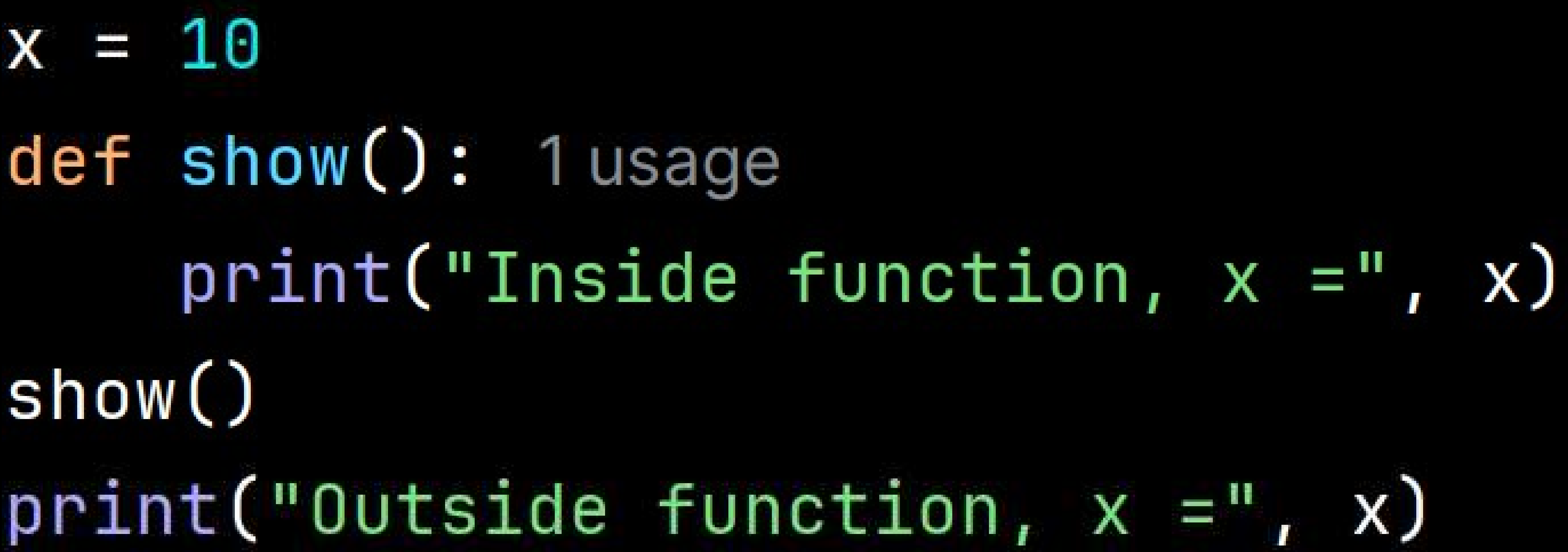
Scope of Variable

Scope of Variable

Scope of Variable

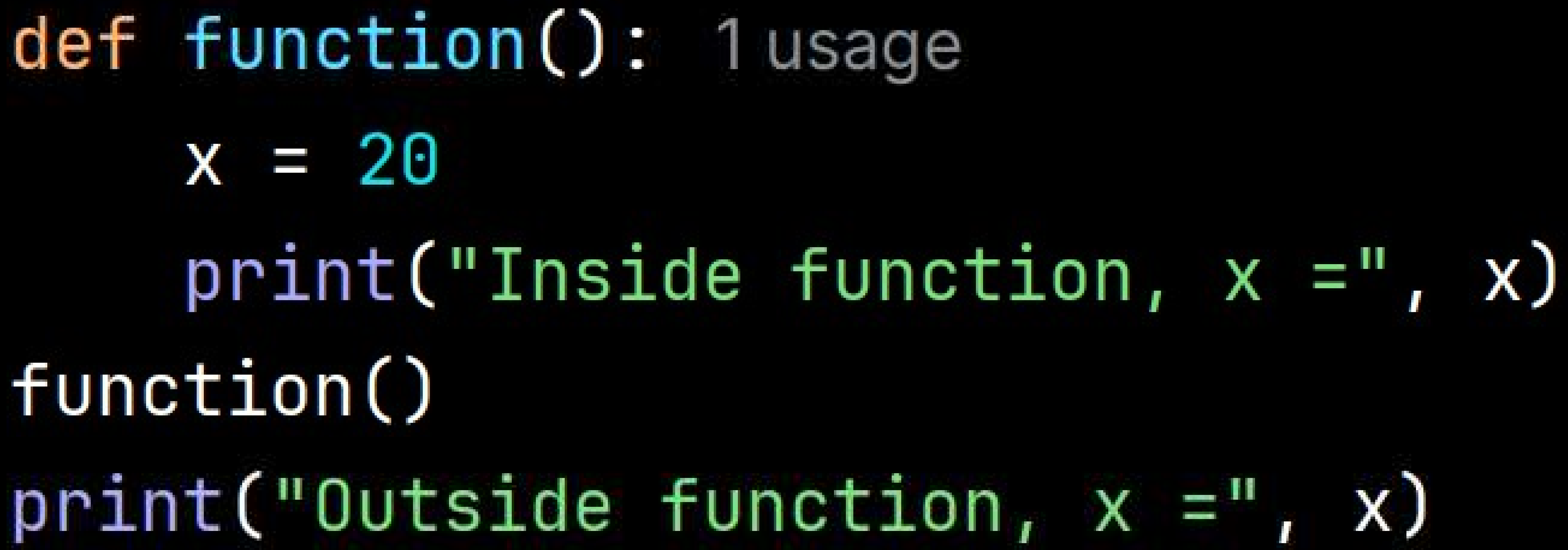
Scope of Variable

028 GLOBAL SCOPE



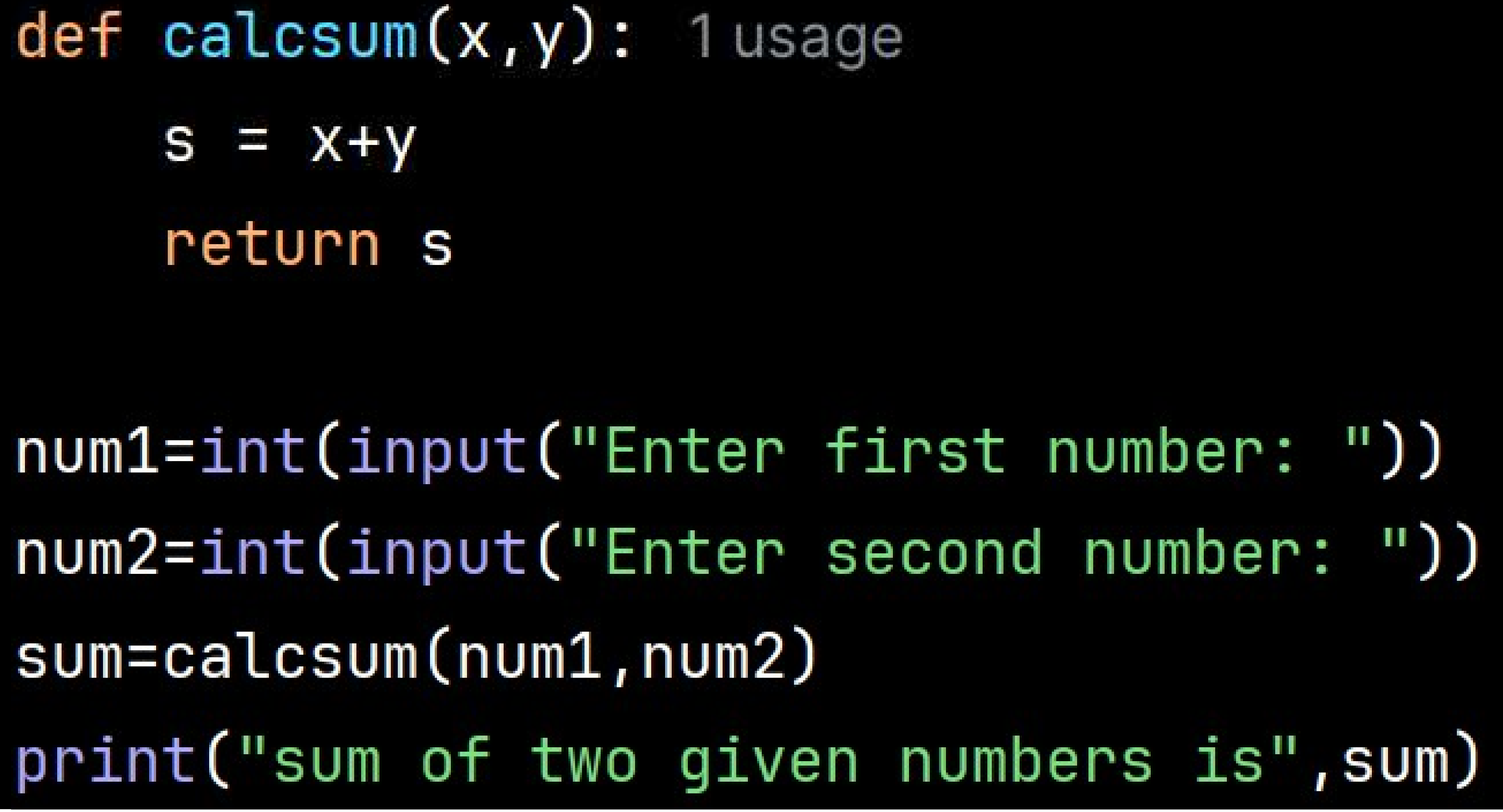
```python
x = 10
def show():
    print("Inside function, x =", x)
show()
print("Outside function, x =", x)
```

029 LOCAL SCOPE



```
def function(): 1 usage
    x = 20
    print("Inside function, x =", x)
function()
print("Outside function, x =", x)
```

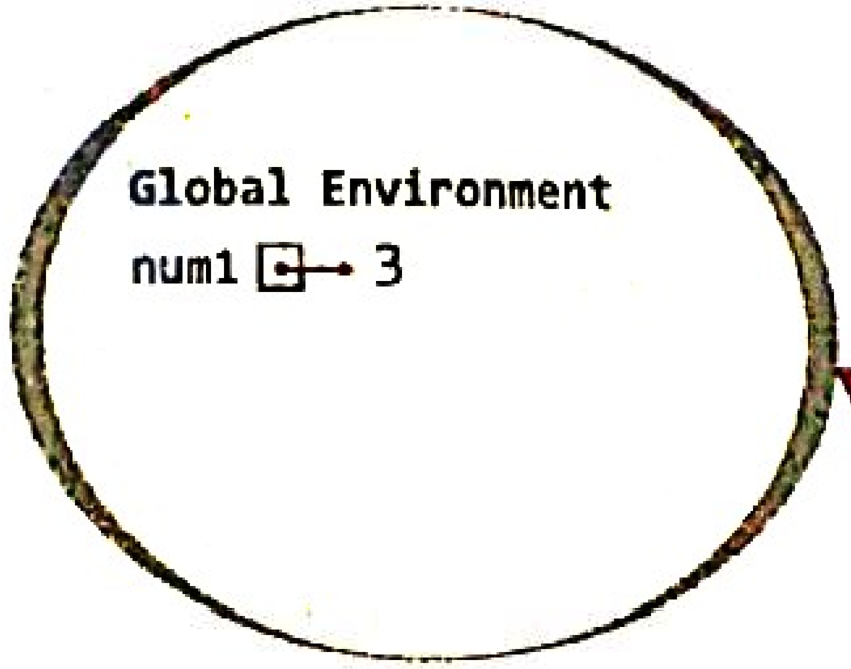
030 SCOPE EXAMPLE 1



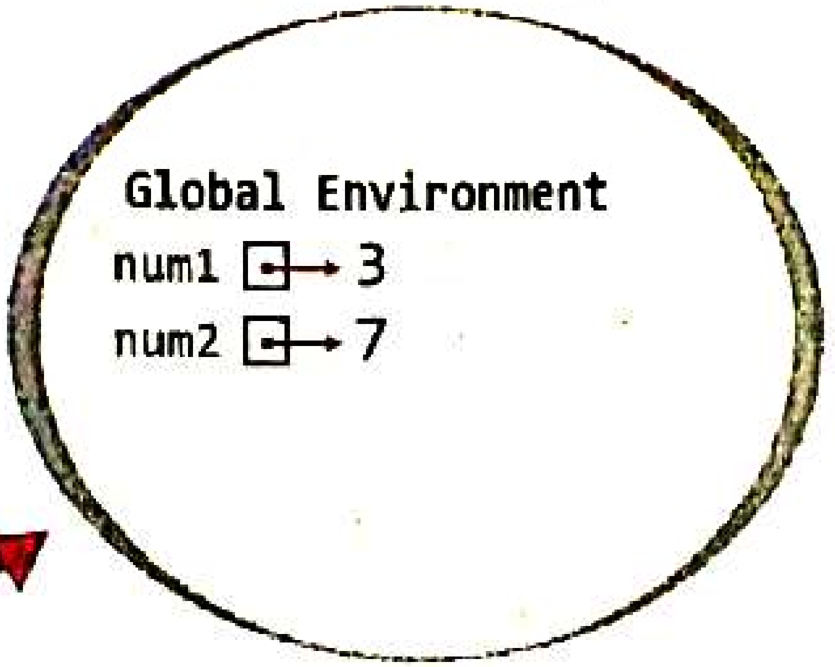
```python
def calcsum(x,y): 1 usage
    s = x+y
    return s


num1=int(input("Enter first number: "))
num2=int(input("Enter second number: "))
sum=calcsum(num1,num2)
print("sum of two given numbers is",sum)
```

Scope of Variable

1. Line1 : *def* encountered and lines 2-3 are ignored.

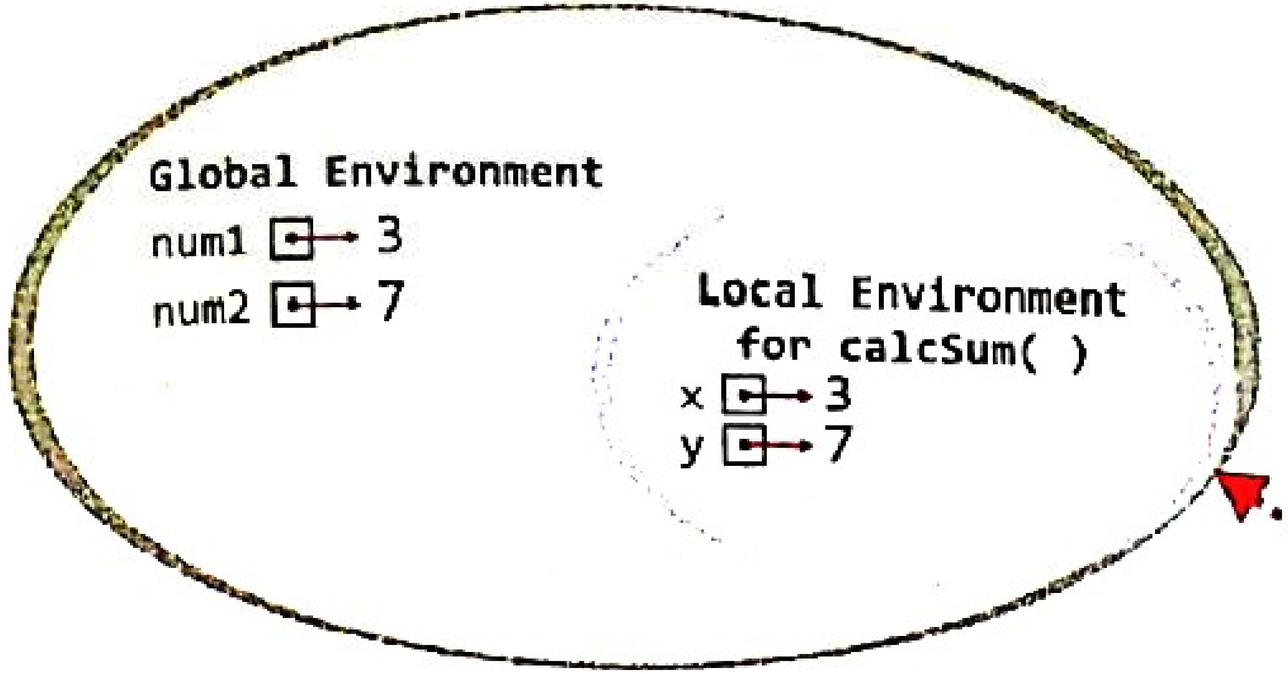


2. Line4 (Main.1) : Execution begins and global environment is created. *num1* is added to this environment.



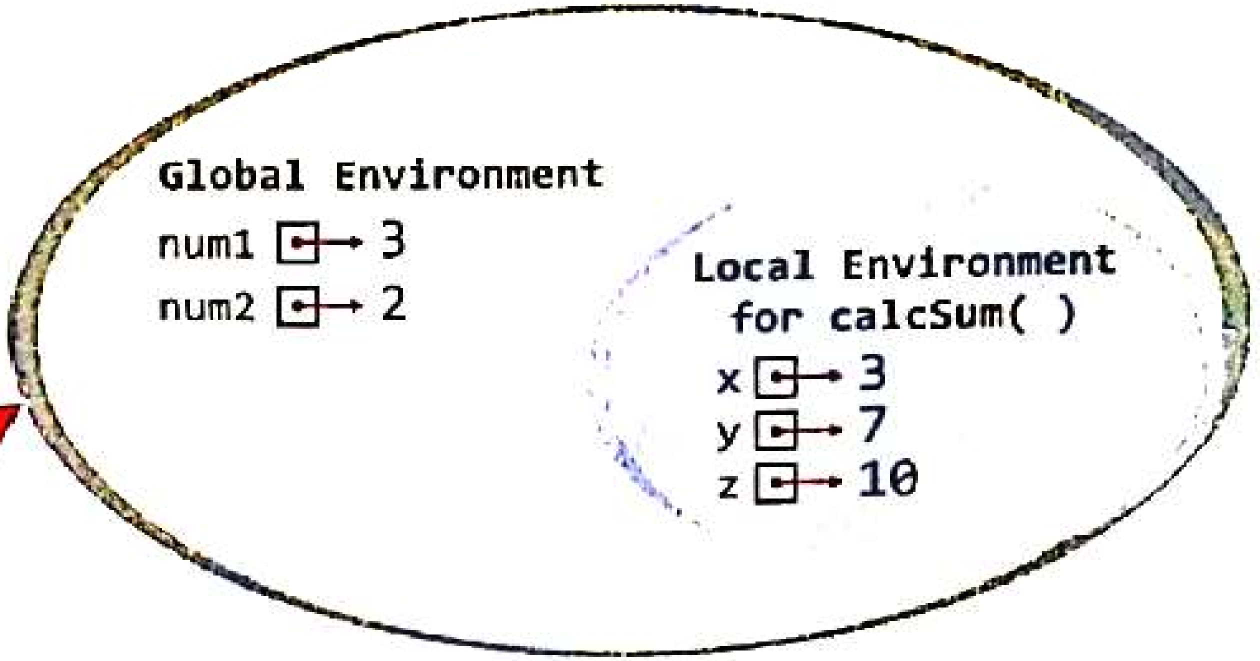
3. Line5 (Main.2) : *num2* is also added to the global environment.

Scope of Variable

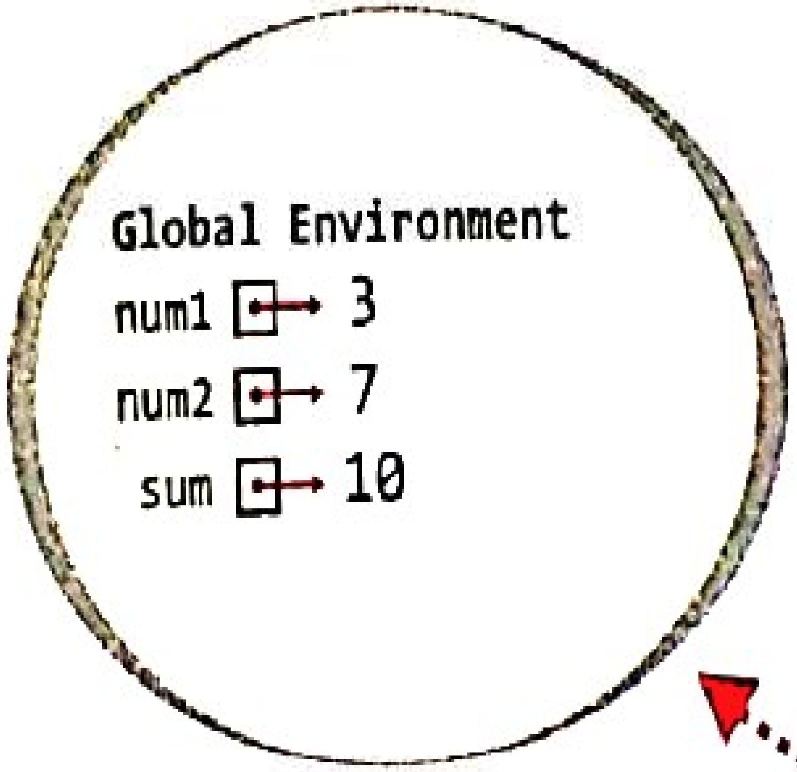


4. Line6 (**Main.3**) : **calcSum( )** is invoked, so a local environment for **calcSum( )** is created; *formal arguments x* and *y* are created in local environment.

5. Line2 (**calcSum.1**) : variable *z* is created in the local environment.



Scope of Variable

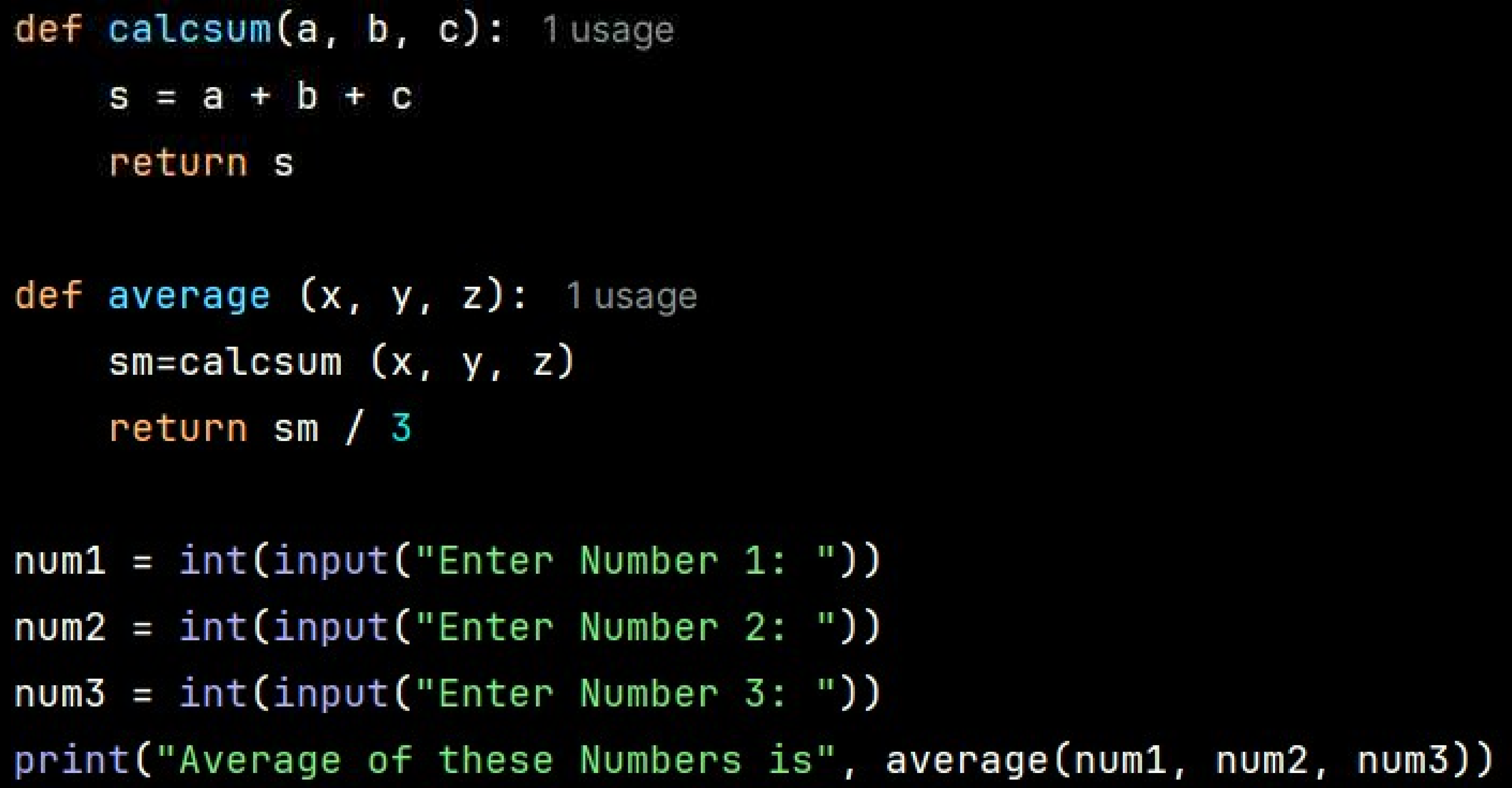


6. Line3 (**calcSum.2**) : value of *z* is returned to caller (**return** ends the function, hence after sending value of *s* to caller in variable *sum* (when control is back to *Main.3*), the local environment is removed and so are all its constituents).

7. Line7 (**Main.4**) : the print statement picks value of *sum* from its own environment.

8. Program over. Global environment is also removed with the end of the program.

031 SCOPE EXAMPLE 2



```python
def calcsum(a, b, c):  1 usage
    s = a + b + c
    return s

def average (x, y, z):  1 usage
    sm=calcsum (x, y, z)
    return sm / 3

num1 = int(input("Enter Number 1: "))
num2 = int(input("Enter Number 2: "))
num3 = int(input("Enter Number 3: "))
print("Average of these Numbers is", average(num1, num2, num3))
```

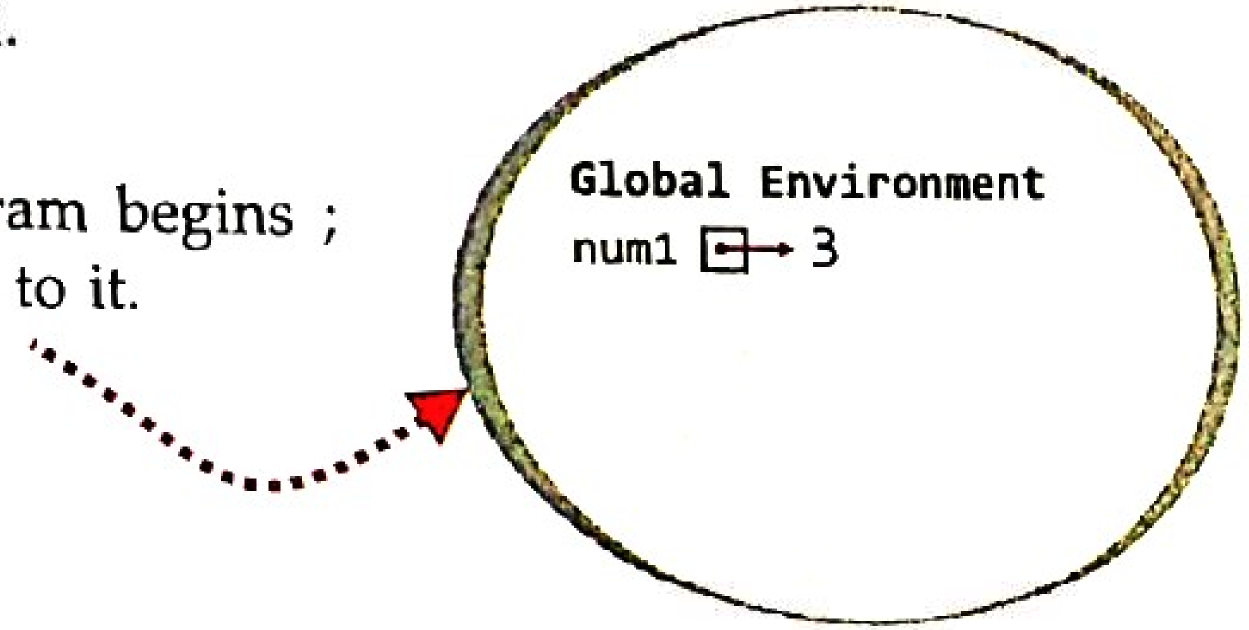
Scope of Variable

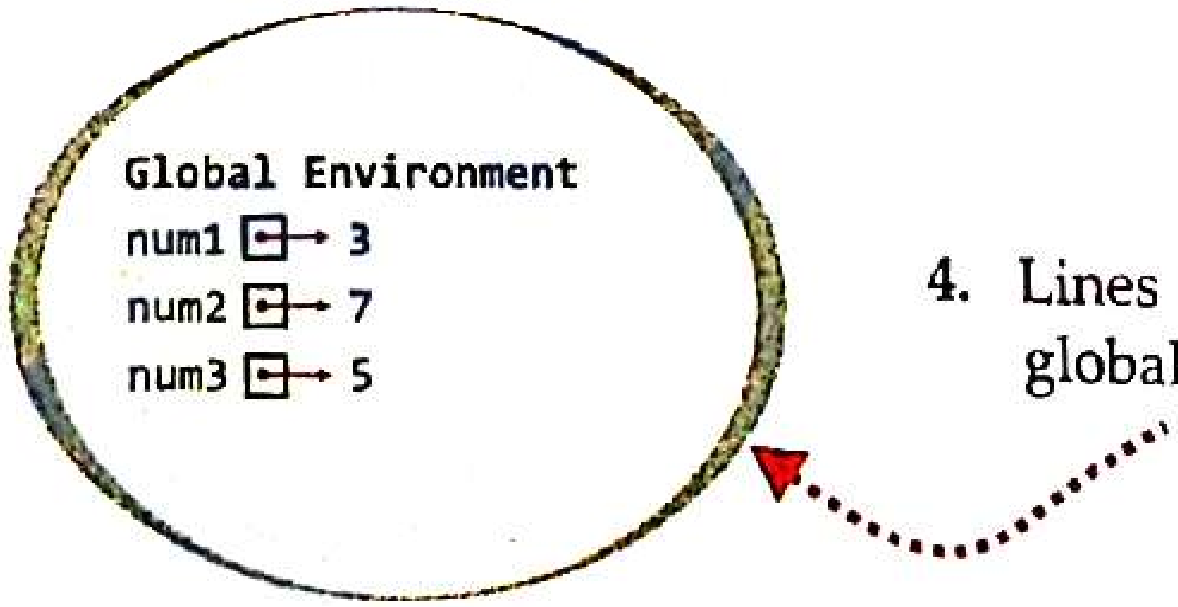
Scope of Variable

1. Line1 : *def* encountered ; lines 2, 3 ignored.

2. Line4 : *def* encountered ; lines 5, 6 ignored.

3. Line7 (**Main.1**) : execution of main program begins ; global environment created ; ***num1*** added to it.

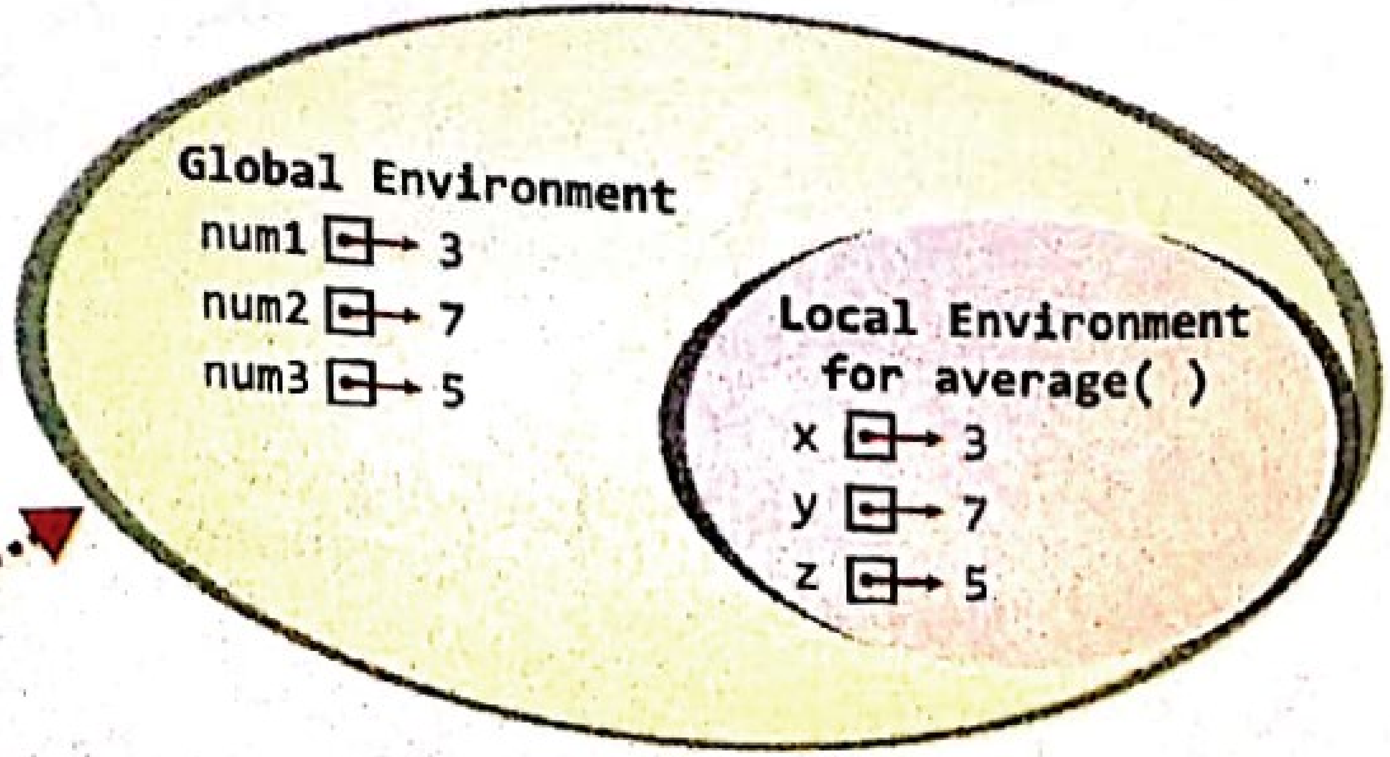




4. Lines 8, 9 (**Main.2** and **Main.3**) : add *num2* and *num3* to global environment.

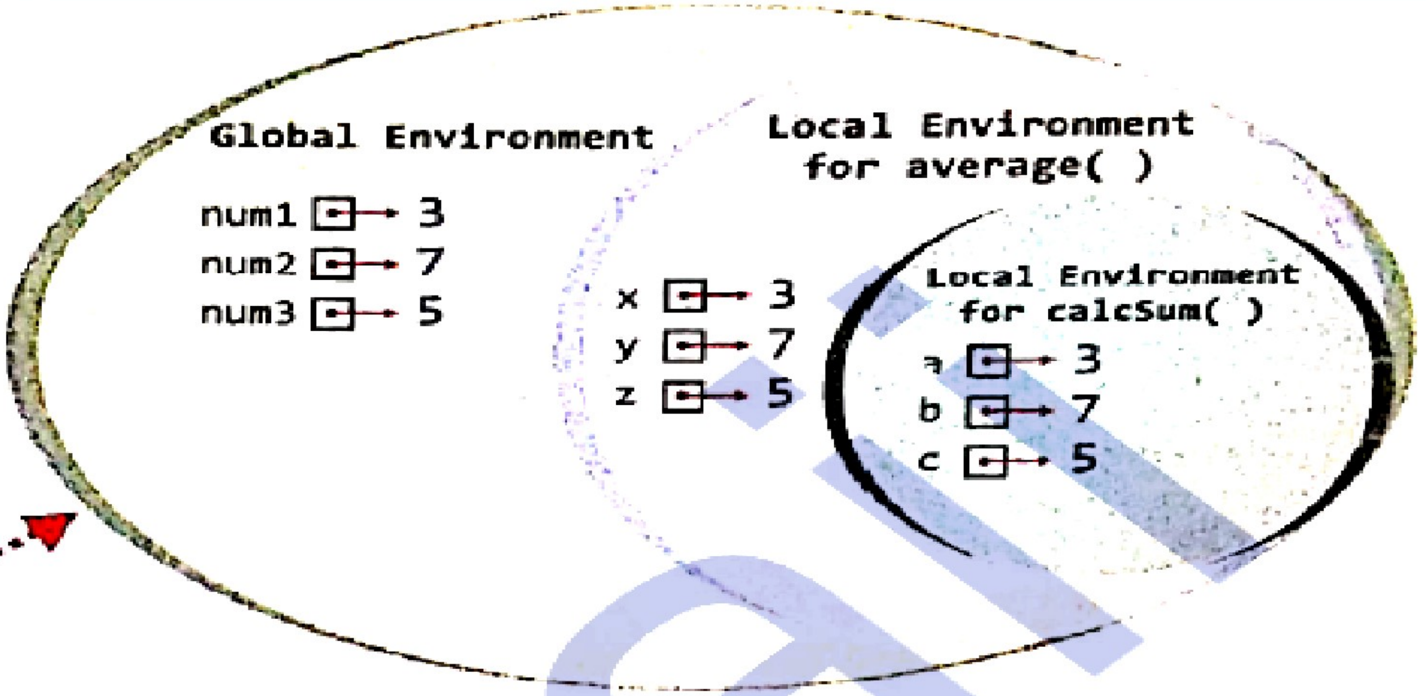
Scope of Variable

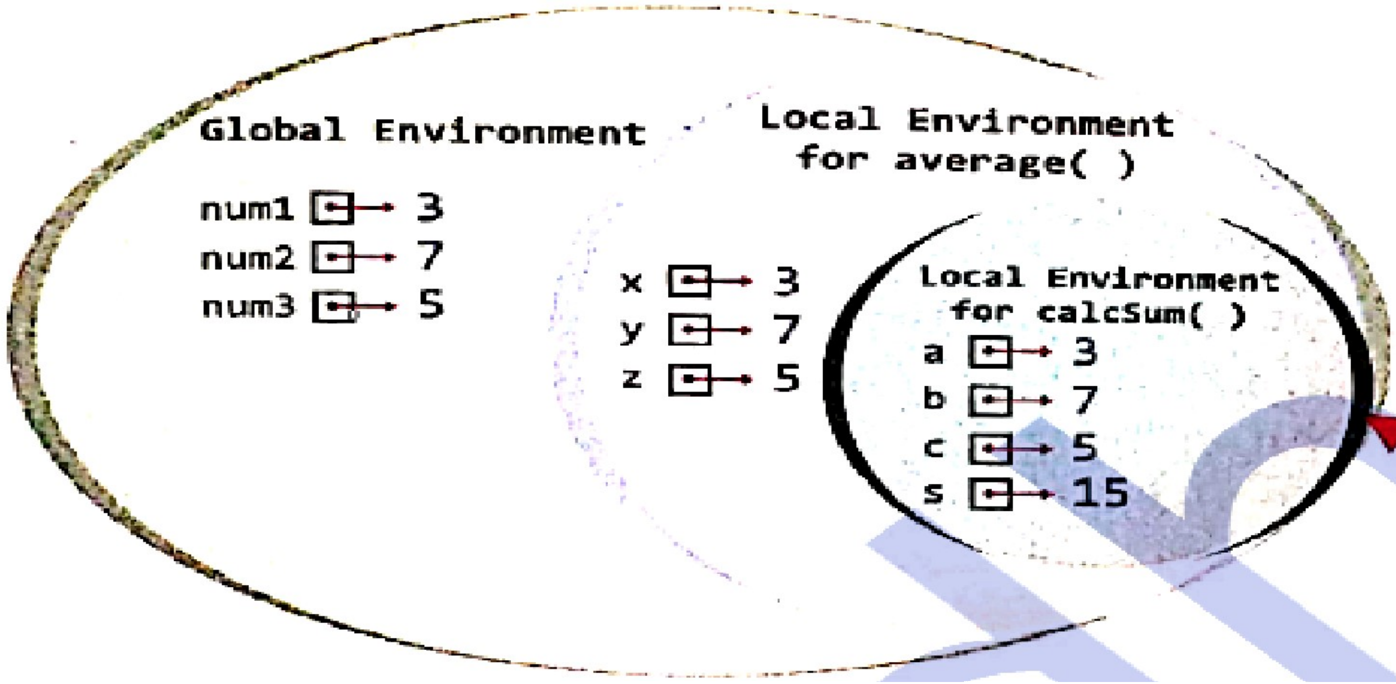
5. Line10 (Main.4) : Function **average( )** is invoked, so a local environment for **average( )** is created ; formal arguments *x*, *y* and *z* are created in local environment.



Scope of Variable

6. Line5 (**average.1**) : Function *calcSum( )* is invoked, so a local environment for *calcSum( )* is created, nested within local environment of *average( )* ; its formal arguments (*a*, *b*, *c*) are created in it.

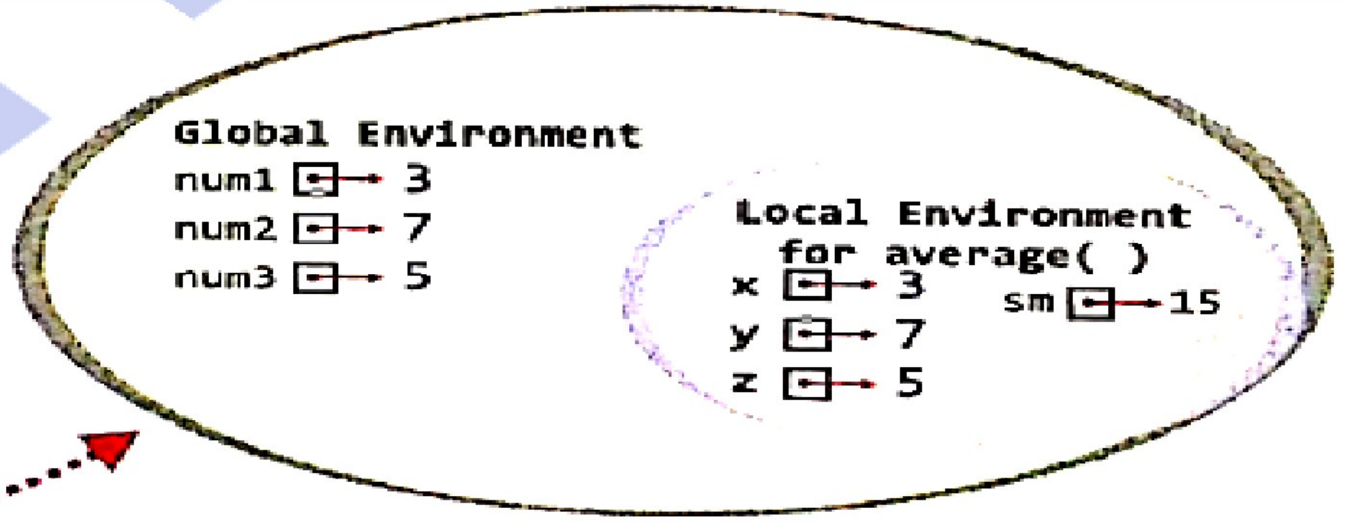




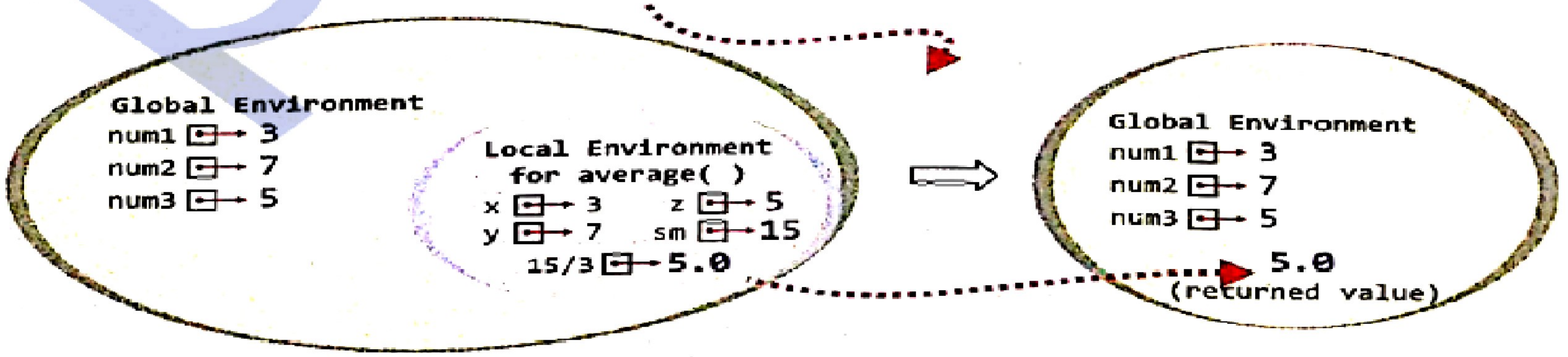
7. Line1 (**calcSum.1**) : Variable *s* is created within local environment of *calcSum( )*.

Scope of Variable

8. Line2 (calcSum.2) : Value of *s* is returned to *sm* of *average*( ) and *calcSum*( ) is over, hence the local environment of *calcSum*( ) is removed.



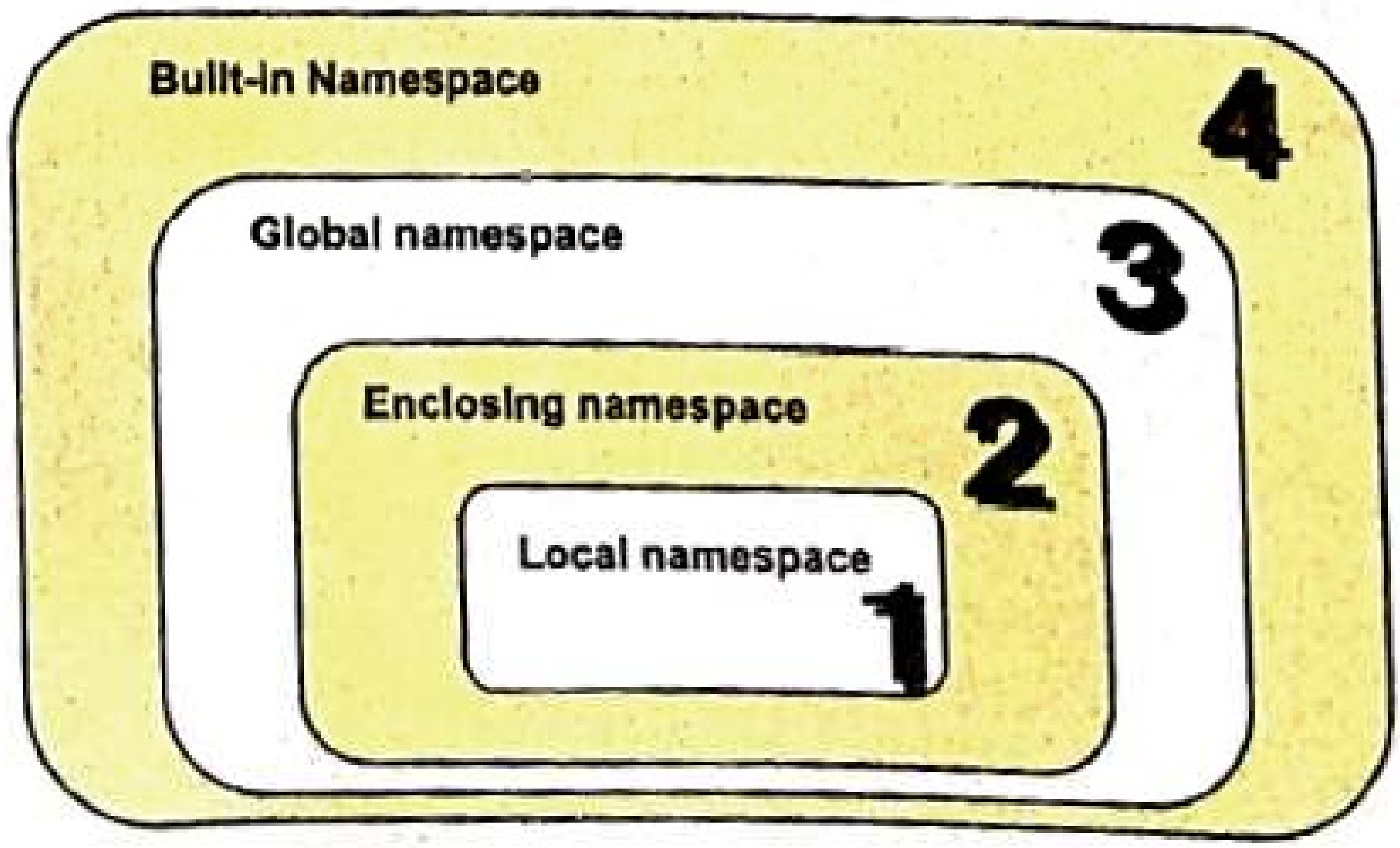
9. Line6 (average.2) : Return value is calculated as *sm* / 3 (*i.e.*, 15/3 = 5.0) and returned to caller (*main.4*) statement ; *average( )* is over so its local environment is removed.



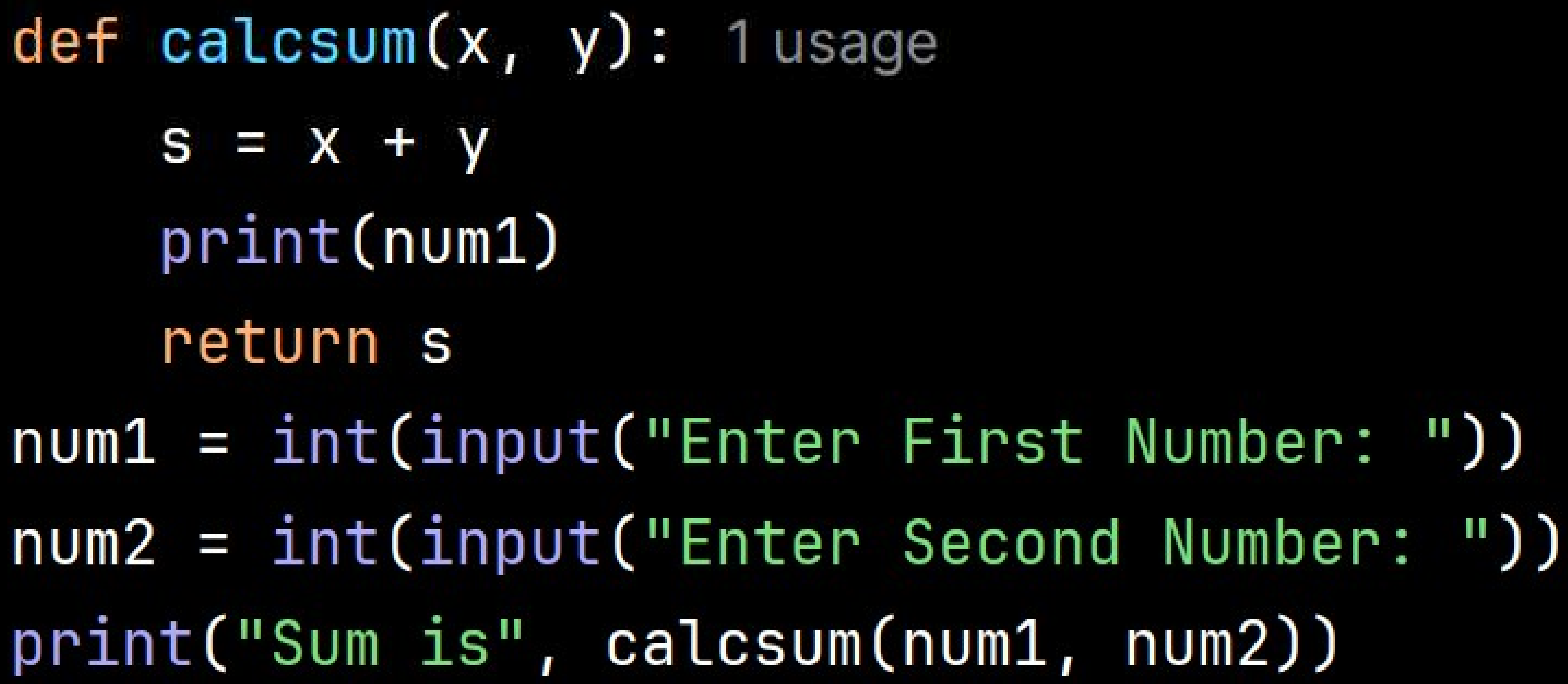
Scope of Variable

Scope of Variable

Scope of Variable



032 CASE 1 GLOBAL BUT NOT LOCAL



```python
def calcsum(x, y):  1 usage
    s = x + y
    print(num1)
    return s
num1 = int(input("Enter First Number: "))
num2 = int(input("Enter Second Number: "))
print("Sum is", calcsum(num1, num2))
```

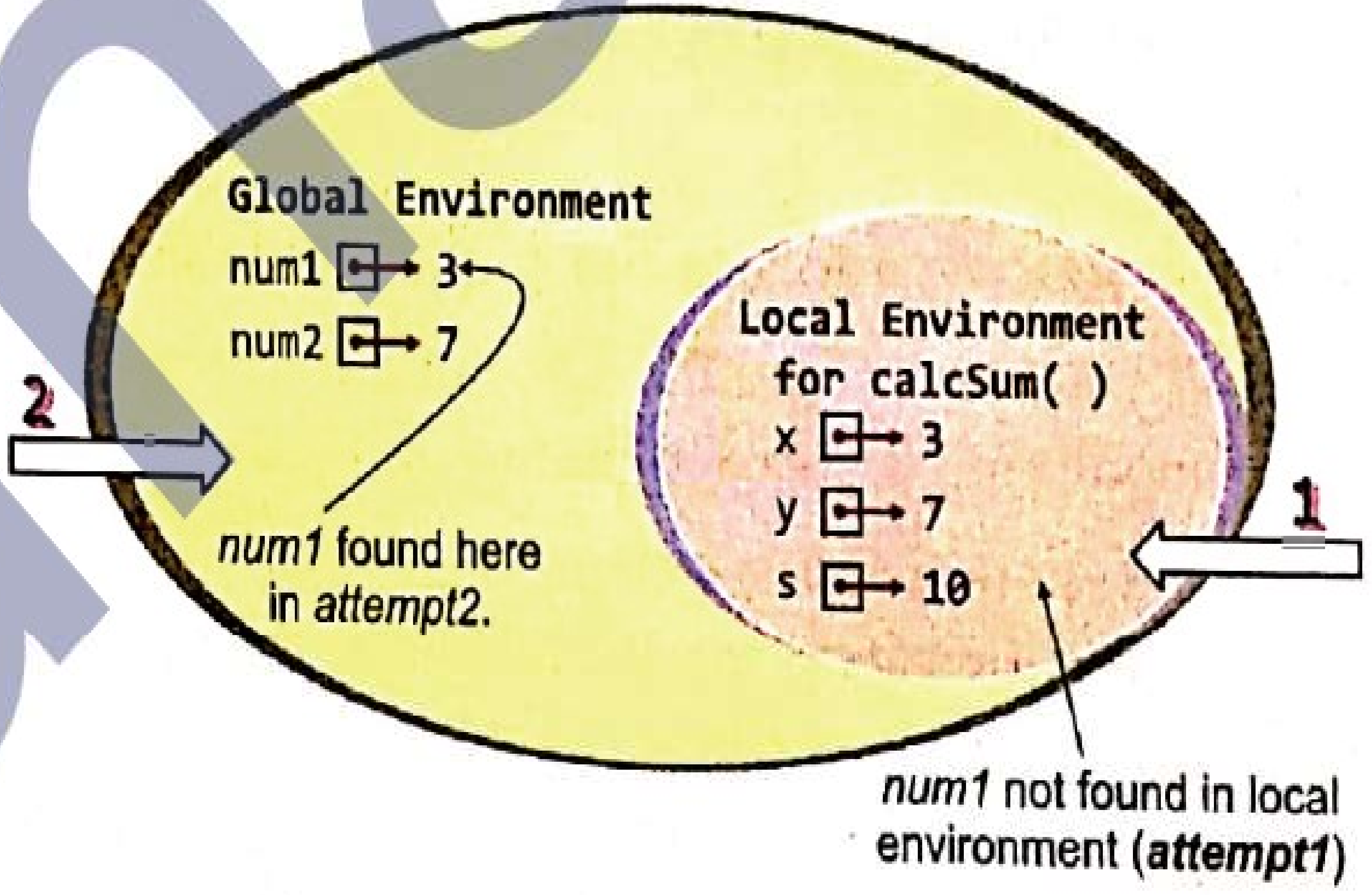
Scope of Variable

1. Python will first check the *Local environment* of ***calcSum( )*** for ***num1*** ;

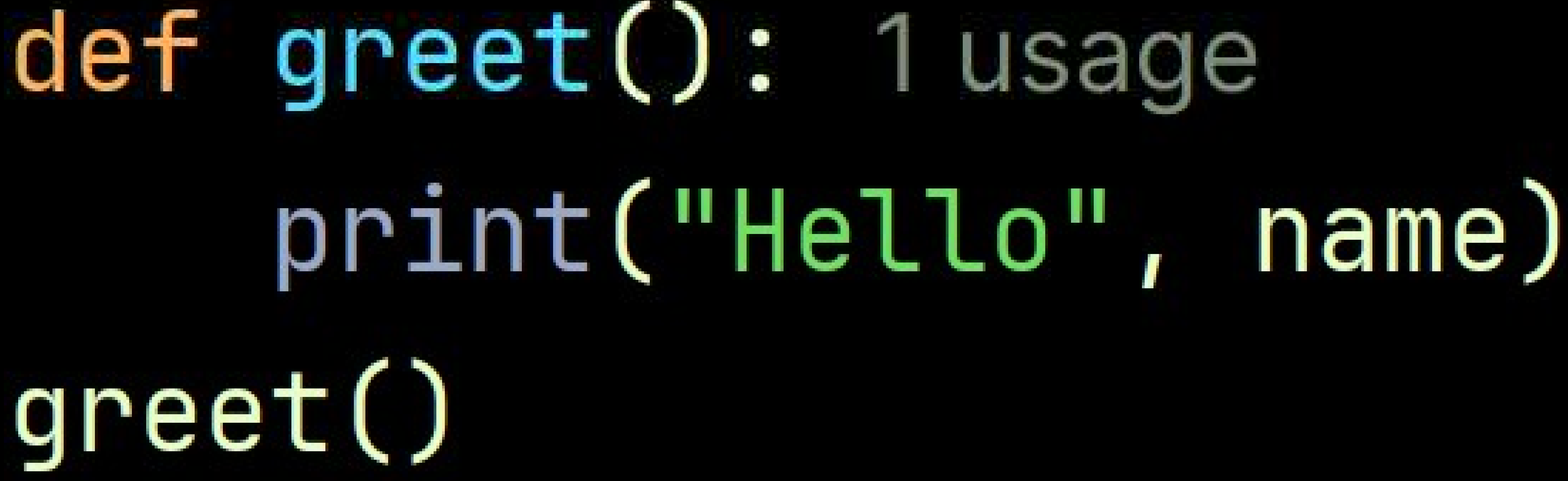
   num1 is not found there.

2. Python now checks for ***num1***, the parent environment of ***calcSum( )***, which is *Global environment* (there is not any intermediate enclosing environment).

   Python finds ***num1*** here ; so it picks its value and prints it.

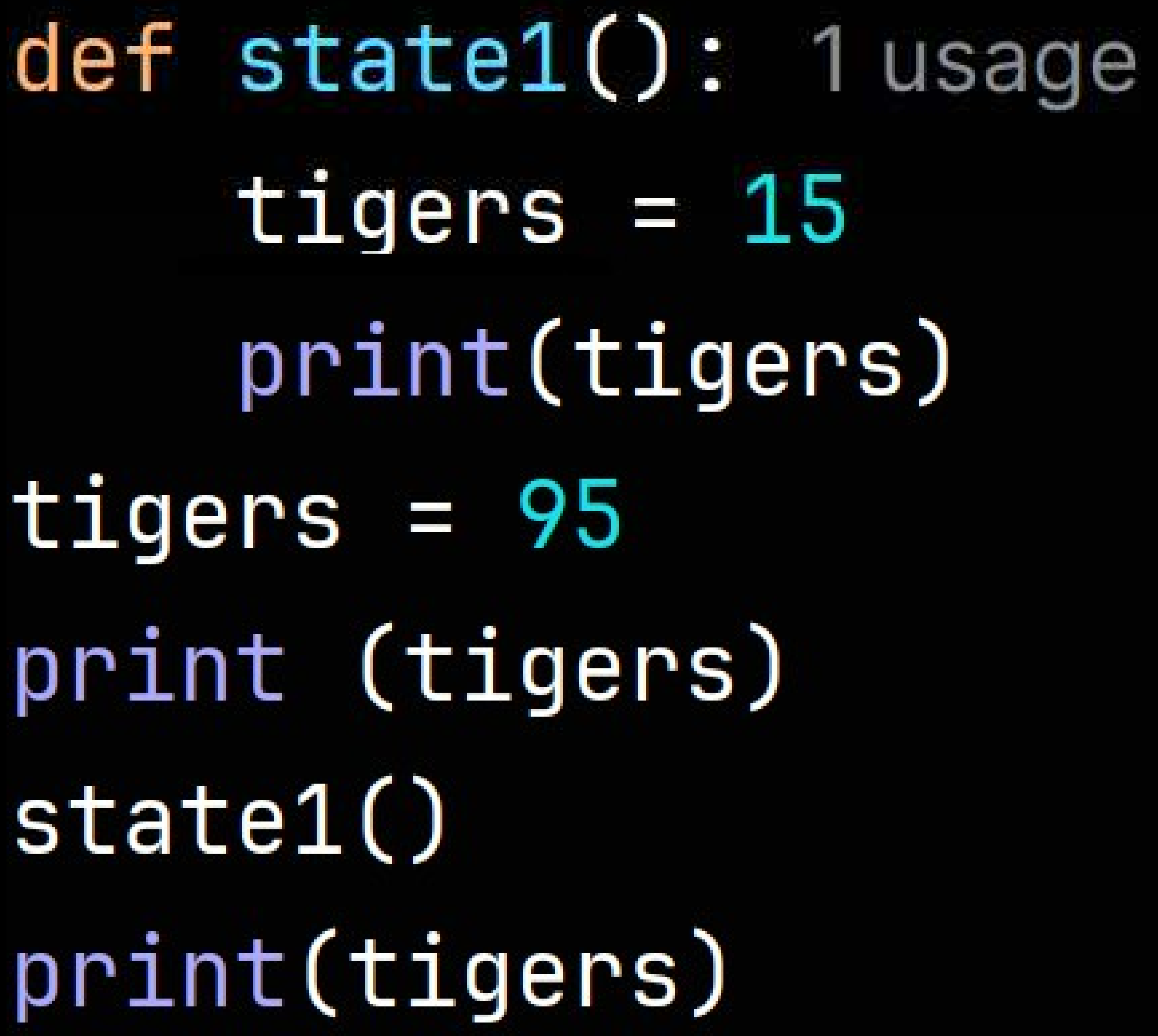

Global Environment
num1 → 3
num2 → 7
Local Environment for calcSum( )
x → 3
y → 7
s → 10
2
1
num1 found here in *attempt2*.
num1 not found in local environment (***attempt1***)

033 CASE 2 NEITHER LOCAL NOR GLOBAL



```python
def greet():  1 usage
    print("Hello", name)
greet()
```

034 CASE 3 BOTH IN GLOBAL AND LOCAL



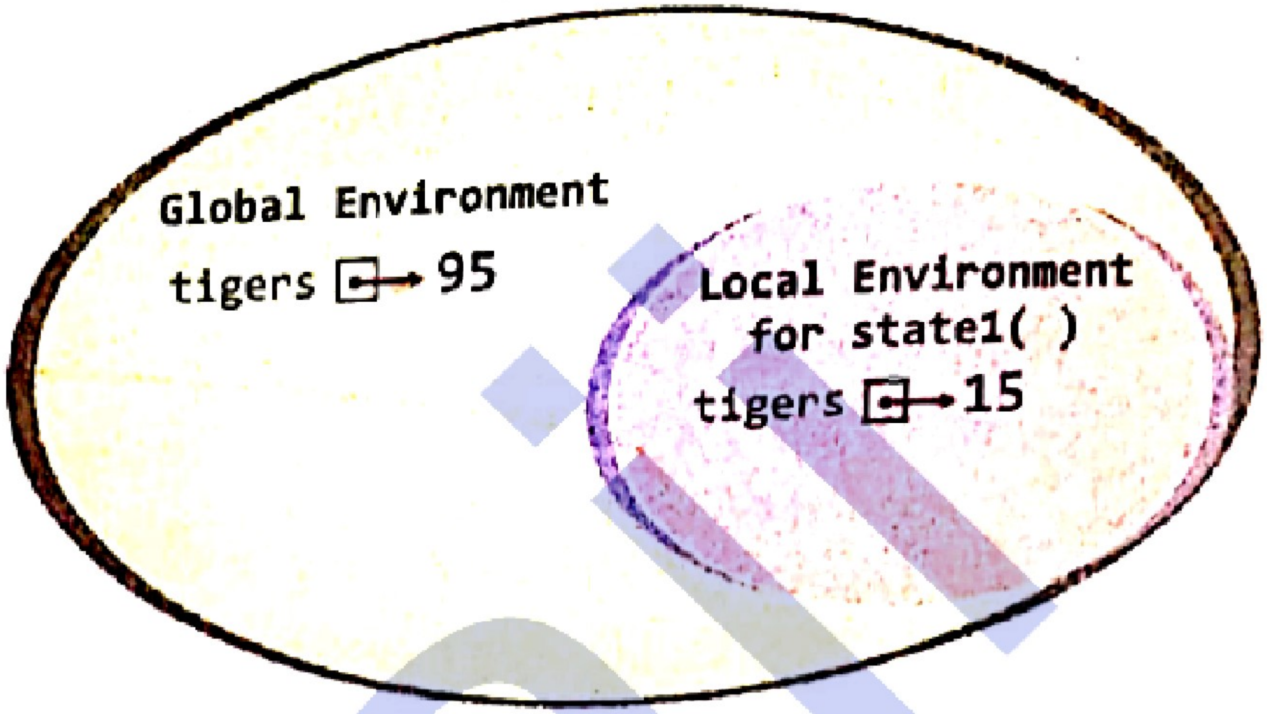
```python
def state1(): 1 usage
    tigers = 15
    print(tigers)
tigers = 95
print (tigers)
state1()
print(tigers)
```

Scope of Variable

Scope of Variable

```
def state1( ) :
    tigers = 15
    print(tigers)

tigers = 95
print tigers
state1()
print(tigers)
```

This statement will create a local variable with name tigers as it is assignment statement. It won't refer to tigers of main program.



The above program will give output as :

```
95
15
95
```

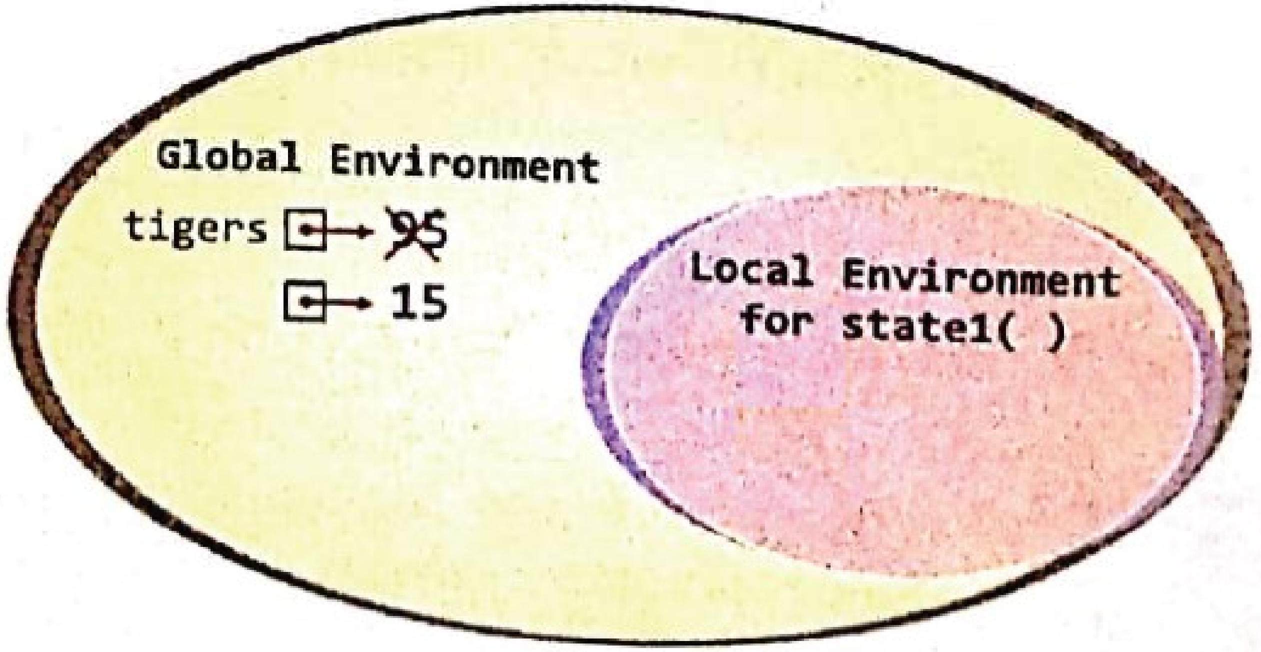
*Result of print statement inside state1( ) function, thus, value of local **tigers** is printed.*

*Result of print statement inside main program, thus, value of global **tigers** is printed.*

Scope of Variable

```
def state1( ) :
    global tigers
    tigers = 15
    print(tigers)

tigers = 95
print(tigers)
state1()
print(tigers)
```

*This is an indication not to create local variable with the name **tigers**, rather use global variable **tigers**.*



The above program will give output as :

```
95
15
15
```

*Result of print statement inside state1( ) function, value of global **tigers** is printed (which was modified to **15** in previous line).*

*Result of print statement inside main program, thus, value of global **tigers** (which is 15 now) is printed.*