Ques. **What is Scope in Python?**

- Every object in Python functions within a scope. A scope is a block of code where an object in Python remains relevant. Namespaces uniquely identify all the objects inside a program.

1. **Local Scope/Local Variables:-** The Variables which are defined in the function are a local scope of the variable. These variables are defined in the function body.

```
x = "awesome"

def myfunc():
    x = "fantastic"
    print("Python is " + x)
myfunc()
print("Python is " + x)

Output:-
Python is fantastic
Python is awesome
```

2. **Global Scope/Global Variables:-** The Variable which can be read from anywhere in the program is known as a global scope. These variables can be accessed inside and outside the function.

```
x = 300
def myfunc():
    print(x)
myfunc()

print(x)
Output:- 300
300
```

3. **NonLocal or Enclosing Scope:-** Nonlocal Variable is the variable that is defined in the nested function. It means the variable can be neither in the local scope nor in the global scope.

```
def func_outer():
    x = "local"
    def func_inner():
        nonlocal x
        x = "nonlocal"
        print("inner:", x)
    func_inner()
    print("outer:", x)
func_outer()

Output:-
```

```
inner: nonlocal
outer: nonlocal
```

4. **Built-in Scope:-** If a Variable is not defined in local, Enclosed or global scope, then python looks for it in the built-in scope. In the Following Example, 1 from math module pi is imported, and the value of pi is not defined in global, local and enclosed. Python then looks for the pi value in the built-in scope and prints the value. Hence the name which is already present in the built-in scope should not be used as an identifier.

```
# Built-in Scope
from math import pi
# pi = 'Not defined in global pi'
def func_outer():
    # pi = 'Not defined in outer pi'
    def inner():
        # pi = 'not defined in inner pi'
        print(pi)
    inner()
func_outer()

Output:- 3.141592
```

Ques. global Keyword?

- To create a global variable inside a function, you can use the global keyword.

```
def myfunc():
  global x
  x = "fantastic"

myfunc()
print("Python is " + x)
Output:- Python is fantastic
```

- Also, use the global keyword if you want to change a global variable inside a function.

```
x = "awesome"
def myfunc():
  global x
  x = "fantastic"

myfunc()
print("Python is " + x)
Output:- Python is fantastic
```