Global Variables

In Python, a variable declared outside of the function or in global scope is known as a global variable. This means that a global variable can be accessed inside or outside of the function.

Let's see an example of how a global variable is created in Python.

Example 1: Create a Global Variable

```
x = "global"

def foo():
    print("x inside:", x)


foo()
print("x outside:", x)
```

Local Variables

A variable declared inside the function's body or in the local scope is known as a local variable.

Example 2: Accessing local variable outside the scope

```
def foo():
    y = "local"


foo()
print(y)
```

Output

```
NameError: name 'y' is not defined
```

The output shows an error because we are trying to access a local variable `y` in a global scope whereas the local variable only works inside `foo()` or local scope.

Example 4: Using Global and Local variables in the same code

```
x = "global "

def foo():
    global x
    y = "local"
    x = x * 2
    print(x)
    print(y)

foo()
```

Output

```
global global
local
```

In the above code, we declare `x` as a global and `y` as a local variable in the `foo()`. Then, we use multiplication operator `*` to modify the global variable `x` and we print both `x` and `y`.

After calling the `foo()`, the value of `x` becomes