1. What is the result of the code, and explain?

>>> X = 'iNeuron'

>>> def func():

print(X)

>>> func()

Output :

iNeuron

x is global variable therefore we are able to print it inside the function , because global variables are accessible inside the functions in python.

2. What is the result of the code, and explain?

>>> X = 'iNeuron'

>>> def func():

X = 'NI!'

>>> func()

>>> print(X)

Output :

iNeuron

we cannot access function variable from outside the function , therefore global variables X get print.

3. What does this code print, and why?

>>> X = 'iNeuron'

>>> def func():

X = 'NI'

print(X)

>>> func()

>>> print(X)

Output :

NI

iNeuron

The global variables are access inside the functions in python. But we cannot access function variable outside function. X is updated with 'NI' inside the function , therefore print function inside the function NI , whereas outside the func() print iNeuron.

4. What output does this code produce? Why?

>>> X = 'iNeuron'

>>> def func():

global X

X = 'NI'

>>> func()

>>> print(X)

Output:

NI

X inside the func() made global , therefore accesible outside of the function too , updated value of X also get printed.

5. What about this code—what’s the output, and why?

>>> X = 'iNeuron'

>>> def func():

X = 'NI'

def nested():

print(X)

nested()

>>> func()

>>> X

Output:

NI

nested() function will print 'iNeuron', but func() does not display anything, x ='NI' is not accessible from outside the func().

6. How about this code: what is its output in Python 3, and explain?

>>> def func():

X = 'NI'

def nested():

nonlocal X

X = 'Spam'

nested()

print(X)

>>> func()

Output:

Spam

The nonlocal variable X inside nested function is not in the local nor the global scope, therefore it print the updated value from nested function.