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

>>> X = 'iNeuron'

>>> def func():

print(X)

>>> func()

Ans: iNeuron

X is global variable and X can be accessed from anywhere.

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

>>> X = 'iNeuron'

>>> def func():

X = 'NI!'

>>> func()

>>> print(X)

Ans: Ineuron

The print function will print the value of X which has been assigned the value in global scope

3. What does this code print, and why?

>>> X = 'iNeuron'

>>> def func():

X = 'NI'

print(X)

>>> func()

>>> print(X)

Ans:

Ni

Ineuron

The value of the variable of x in local scope within the function is NI, that will be printed first

And with the following print statement, Ineuron will be printed

4. What output does this code produce? Why?

>>> X = 'iNeuron'

>>> def func():

global X

X = 'NI'

>>> func()

>>> print(X)

Ans:

As we are using the global keyword here, the value of X will be changed to Ni and the same value (Ni) will be 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

Ans:

Ni

Ineuron

First the nested function will print the value of x defined in function fun which is equal to ‘Ni’

Then the value in the global scope will be printed.

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()

Ans: Spam

Nonlocal variables are used in nested functions whose local scope is not defined. This means that the variable can be neither in the local nor the global scope.

In this case nested function has marked the variable as non local variable therefore, it will be able to make modification to the variable X and will print Spam