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

>>> X = 'iNeuron'

>>> def func():

print(X)

>>> func()

Ans => iNeuron because we have assigned iNeuron string to variable x and printing it inside our function

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

>>> X = 'iNeuron'

>>> def func():

X = 'NI!'

>>> func()

>>> print(X)

Ans => iNeuron will print because we are printing x and not returning or priting any value in our function

3. What does this code print, and why?

>>> X = 'iNeuron'

>>> def func():

X = 'NI'

print(X)

>>> func()

>>> print(X)

Ans => it will print iNeuron two times because we are printing it two times and not returning any value from function

4. What output does this code produce? Why?

>>> X = 'iNeuron'

>>> def func():

global X

X = 'NI'

>>> func()

>>> print(X)

Ans => it will print NI because global keyword refers to global scope of variable and because of this value NI will be overridden in variable from iNeuron

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=> nothing will print because we are only calling function func and it is not returning anything we are only placing x not printing

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 =>

SyntaxError: no binding for nonlocal 'X' found

Because we are assigning value to variable x after definition of function we have to define it before it.