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

>>> X = 'iNeuron'

>>> def func():

print(X)

>>> func()

iNeuron

because function has a defination print

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

>>> X = 'iNeuron'

>>> def func():

X = 'NI!'

>>> func()

>>> print(X)

None

doesnt have a return type

3. What does this code print, and why?

>>> X = 'iNeuron'

>>> def func():

X = 'NI'

print(X)

>>> func()

>>> print(X)

NI

iNeuron

func() has a local variable, print(X) is global variable

4. What output does this code produce? Why?

>>> X = 'iNeuron'

>>> def func():

global X

X = 'NI'

>>> func()

>>> print(X)

NI

func() doesnt have any value to return

whereas X is a global variables value has been changed to NI because it is in the local definition

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

>>> X = 'iNeuron'

>>> def func():

X = 'NI'

def nested():

print(X)

nested()

>>> func()

>>> X

NI

iNeuron

when func() gets called the print statement in the nested function gets called and find the value of X

has NI

X is a global variable

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()

Spam

the answer would have been NI if the keyword nonlocal would not be defined therefore the variable

X doesnt belong to the func() in local scope