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

>>> X = 'iNeuron'

>>> def func():

print(X)

>>> func()

ANS: The global variables are accessible in side the functions in python. But we can not access function variable out side

function.

Since x is golbal variable we are able to print it in side the function

solution : 'iNeuron'

In [12]:

X **=** 'iNeuron'

**def** func():

print(X)

func()

iNeuron

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

>>> X = 'iNeuron'

>>> def func():

X = 'NI!'

>>> func()

>>> print(X)

ANS: The global variables are access in side the functions in python. But we can not access function variable out side

function.

Since x is golbal variable we are able to print it out side of the function

solution = 'iNeuron'

In [13]:

X **=** 'iNeuron'

**def** func():

X **=** 'NI!'

func()

print(X)

iNeuron

3. What does this code print, and why?

>>> X = 'iNeuron'

>>> def func():

X = 'NI'

print(X)

>>> func()

>>> print(X)

ANS: The global variables are access in side the functions in python. But we can not access function variable out side

function. X is updated with 'NI' which is local to function and its immutable. its name space is with in the function

solution = 'NI!', 'iNeuron'

In [15]:

X **=** 'iNeuron'

**def** func():

X **=** 'NI!'

print(X)

func()

print(X)

NI!

iNeuron

4. What output does this code produce? Why?

>>> X = 'iNeuron'

>>> def func():

global X

X = 'NI'

>>> func()

>>> print(X)

ANS: since the X in side function is made Global, it will be accesible out side of the function too. now X will have new value.

solution : 'NI!', 'NI!'

In [18]:

X **=** 'iNeuron'

**def** func():

**global** X

X **=** 'NI!'

print(X)

func()

print(X)

NI!

NI!

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: the nested() function will print 'iNeuron', Then func() does not display anything, and x ='NI' is not accessible out

side the function.

Solution : 'iNeuron'

In [26]:

X **=** 'iNeuron'

**def** func():

X **=** 'NI'

**def** nested():

print(X)

nested()

func()

X

iNeuron

Out[26]:

'iNeuron'

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: 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. it print the updated value from nested

function

Solution : 'spam'

In [44]:

**def** func():

X **=** 'NI'

**def** nested():

**nonlocal** X

X **=** 'spam'

nested()

print(X)

func()

spam