**Yashwant Desai – Assignment 22**

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

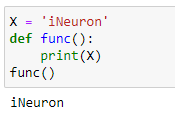
>>> X = 'iNeuron'

>>> def func():

print(X)

>>> func()

Answer: Above code will give “expected an indented block” error due to the code indentation is incorrect in the func() definition. Below is corrected code.



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

>>> X = 'iNeuron'

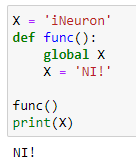
>>> def func():

X = 'NI!'

>>> func()

>>> print(X)

Answer: Above code will give “expected an indented block” error due to the code indentation is incorrect in the func definition also the variable name is same in function and local. To avoid confusion and potential errors it is a good practice to use different names for global and local variables. Below is corrected code.



3. What does this code print, and why?

>>> X = 'iNeuron'

>>> def func():

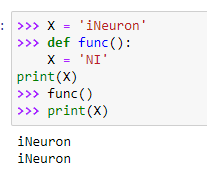
X = 'NI'

print(X)

>>> func()

>>> print(X)

Answer: Above code will give “expected an indented block” error due to the code indentation is incorrect in the func definition. If we fix indentation code will give us “iNeuron” result. The function func creates a new local variable X within its scope and any changes made to this local variable do not affect the global variable X. Below is the code



4. What output does this code produce? Why?

>>> X = 'iNeuron'

>>> def func():

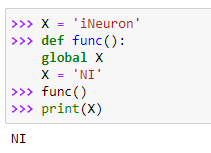
global X

X = 'NI'

>>> func()

>>> print(X)

Answer: If we fix function indentation then we will get “NI” code output as we have defined global variable in the function. Below is an updated code.



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

>>> X = 'iNeuron'

>>> def func():

X = 'NI'

def nested():

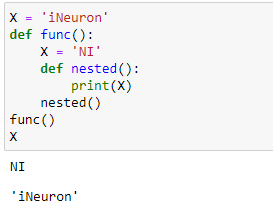
print(X)

nested()

>>> func()

>>> X

Answer: If we fix an indentation issue output of above code will be NI 'iNeuron'. The local variable X defined within the func function does not affect the global variable X. The global variable X retains its original value 'iNeuron'. X = 'NI'. This creates a new local variable X within the scope of the func() function. Below is an updated code



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

>>> def func():

X = 'NI'

def nested():

nonlocal X

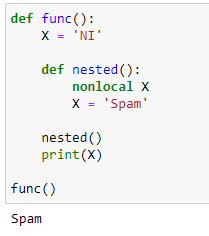
X = 'Spam'

nested()

print(X)

>>> func()

Answer: If we fix an indentation issue output of above code will be Spam. The nonlocal keyword allows us to modify variables in an enclosing (non-global) scope, and in this case it allows the nested function to modify the local variable X in the func() function. Below is an updated code.



**Regards,**

**Yashwant**