PYTHON ASSIGNMENT – 22

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

>>> X = 'iNeuron'

>>> def func():

print(X)

>>> func()

The function func() doesn't define a local variable X, so it accesses the global variable X, which is assigned the value 'iNeuron'. When the function is called, it prints the value of the global variable X.

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

>>> X = 'iNeuron'

>>> def func():

X = 'NI!'

>>> func()

>>> print(X)

Inside the function func(), a local variable X is defined and assigned the value 'NI!'. However, this variable is local to the function and does not affect the global variable X. When the function is called, it doesn't print anything. Then, when print(X) is called outside the function, it prints the value of the global variable X, which remains 'iNeuron'.

3. What does this code print, and why?

>>> X = 'iNeuron'

>>> def func():

X = 'NI'

print(X)

>>> func()

>>> print(X)

NI

iNeuron

Inside the function func(), a local variable X is defined and assigned the value 'NI'. When the function is called, it prints the value of the local variable X. Outside the function, when print(X) is called, it prints the value of the global variable X, which remains 'iNeuron'.

4. What output does this code produce? Why?

>>> X = 'iNeuron'

>>> def func():

global X

X = 'NI'

>>> func()

>>> print(X)

NI

Inside the function func(), the global keyword is used to declare that X refers to the global variable X. Then, X is assigned the value 'NI'. When the function is called, it modifies the global variable X. Thus, when print(X) is called outside the function, it prints the updated value of X.

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

Inside the function func(), a local variable X is defined and assigned the value 'NI'. Then, a nested function nested() is defined, which prints the value of X. When func() is called, it executes the nested function nested(), which prints the value of the local variable X. Outside the function, X retains its global value '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()

The code produces a SyntaxError: no binding for nonlocal 'X' found

The nonlocal keyword is used to indicate that the variable X should refer to the nearest enclosing scope, which in this case is the outer function func(). However, since there is no outer scope containing X, Python raises a SyntaxError. This code would work in Python 3 if X were defined in an enclosing scope.