PYTHON ASSIGNMENT - 23

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

>>> def func(a, b=6, c=8):

print(a, b, c)

>>> func(1, 2)

1 2 8

The function func is called with two arguments, 1 and 2. The first argument 1 is assigned to parameter a, and the second argument 2 is assigned to parameter b. The default value for parameter c is 8, so it remains 8.

2. What is the result of this code, and why?

>>> def func(a, b, c=5):

print(a, b, c)

>>> func(1, c=3, b=2)

1 2 3

Explanation: The function func is called with three arguments: positional argument 1 for a, and keyword arguments c=3 and b=2. The keyword arguments override the default values, so a is 1, b is 2, and c is 3.

3. How about this code: what is its result, and why?

>>> def func(a, \*pargs):

print(a, pargs)

>>> func(1, 2, 3)

1 (2, 3)

The function func is called with three arguments: 1, 2, and 3. The first argument 1 is assigned to parameter a. The remaining arguments 2 and 3 are collected into a tuple and assigned to pargs.

4. What does this code print, and why?

>>> def func(a, \*\*kargs):

print(a, kargs)

>>> func(a=1, c=3, b=2)

1 {'c': 3, 'b': 2}

The function func is called with keyword arguments a=1, c=3, and b=2. The argument a=1 is assigned to parameter a. The remaining keyword arguments c=3 and b=2 are collected into a dictionary and assigned to kargs.

5. What gets printed by this, and explain?

>>> def func(a, b, c=8, d=5): print(a, b, c, d)

>>> func(1, \*(5, 6))

1 5 6 5

The function func is called with positional argument 1 for a, and the tuple (5, 6) is unpacked into arguments b and c. The default value for d remains 5, so a is 1, b is 5, c is 6, and d is 5.

6. what is the result of this, and explain?

>>> def func(a, b, c): a = 2; b[0] = 'x'; c['a'] = 'y'

>>> l=1; m=[1]; n={'a':0}

>>> func(l, m, n)

>>> l, m, n

(1, ['x'], {'a': 'y'})

The function func modifies its arguments. The line a = 2 reassigns a locally within the function, but doesn't affect the global variable l. The line b[0] = 'x' modifies the first element of the list m, changing it to 'x'. The line c['a'] = 'y' modifies the dictionary n, changing the value associated with the key 'a' to 'y'. After the function call, l remains 1, m is modified to ['x'], and n is modified to {'a': 'y'}.