### What is the result of the code, and why?

python

Copy code

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

print(a, b, c)

>>> func(1, 2)

**Output:**

Copy code

1 2 8

**Explanation:**

* func is defined with default values for b and c. When func(1, 2) is called, a is set to 1 and b is explicitly set to 2. Since c is not provided in the call, it uses the default value 8.

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

python

Copy code

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

print(a, b, c)

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

**Output:**

Copy code

1 2 3

**Explanation:**

* In this function call, a is set to 1, b is explicitly set to 2, and c is explicitly set to 3. The parameters are assigned in the order they are specified in the function call, so a=1, b=2, and c=3.

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

python

Copy code

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

print(a, pargs)

>>> func(1, 2, 3)

**Output:**

scss

Copy code

1 (2, 3)

**Explanation:**

* \*pargs collects any additional positional arguments into a tuple. Here, a is 1 and \*pargs collects 2 and 3 into the tuple (2, 3).

### 4. What does this code print, and why?

python

Copy code

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

print(a, kargs)

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

**Output:**

arduino

Copy code

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

**Explanation:**

* \*\*kargs collects keyword arguments into a dictionary. Here, a is 1 and the rest of the keyword arguments c=3 and b=2 are collected into the dictionary {'c': 3, 'b': 2}.

### 5. What gets printed by this, and explain?

python

Copy code

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

print(a, b, c, d)

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

**Output:**

Copy code

1 5 6 5

**Explanation:**

* The \* operator unpacks the tuple (5, 6) into positional arguments. Here, a is 1, b is 5, and c and d take the values 6 and 5 respectively (with d using its default value since there are no more values to unpack).

### 6. What is the result of this, and explain?

python

Copy code

>>> 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

**Output:**

css

Copy code

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

**Explanation:**

* a is assigned the value 2 inside the function, but this does not affect the original l because l is an integer and integers are immutable.
* b[0] = 'x' modifies the list m, so m becomes ['x'].
* c['a'] = 'y' modifies the dictionary n, setting the value of key 'a' to 'y'. So n becomes {'a': 'y'}.