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

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

print(a, b, c)

>>> func(1, 2)

Ans:

1 2 8

Because, value passed to a is 1, b is 2 and default value of c is 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)

Ans:

1 2 3

Because, value passed to a is 1, b is 2 and value passed to 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)

Ans:

1 (2, 3)

Because value passed to a is 1 and \*pargs allows to pass any number of extra arguments.

4. What does this code print, and why?

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

print(a, kargs)

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

Ans:

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

Because value passed to a is 1 and \*\*kargs allows to pass **keyworded** any number of arguments to a function.

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))

Ans:

1 5 6 5

Because value passed to a is 1, b is 5, c is 6 and default value of 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

Ans:

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

Because value of l is 1 and it will not change whereas list m and dictionary n are mutable

and reference to same object inside the function and hence they are modified.