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

1 will be assigned to the first positional arg a and 2 will be overwrite the b’s default value and c will take the default value as 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

A will take the passed value as 1 b & c will overwrite the passed values as 2,3 and will print those

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)

Here the 1 will be assigned to a and 2,3 will be assigned to the \*pargs (which can take multiple input) and will print in the form of tuple.

4. What does this code print, and why?

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

print(a, kargs)

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

Output:

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

Here a will be assigned with the value 1 and \*\*kargs will take an input as a dictionary so the output will be reinterpreted as c,b in the form of dictionary.

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

Here a will take the passed input 1 and b&c will take the input passed as \*args 5&6 and overwrite the default value. Whereas d will print the default values.

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'})

Here to the function we are passing the l,m,n

So l will overwrite the values of a, for m and n the values are getting assigned in the function when called so they are respectively as x and {‘a’:’y’}