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

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

print(a, b, c)

>>> func(1, 2)

the value is ‘1 2 8’ because the 1 and 2 was passed in a,b respestively as the default value of c was 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)

the result is ‘ 1 2 3’ the value of a position was passed where the value of b and c was as with reverse though it doesnt matter when passed as key agrument

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

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

print(a, pargs)

>>> func(1, 2, 3)

the result is ‘ 1 (2,3)’ , 1 was in the a position whereas the all other value was collected as positional argument i.e \*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:3}’ is the value , 1 was in the a position whereas the all other value was collected as key argument i.e \*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’ is the result

1 was in the a position , b , c was overwritten with the value 5 , 6 and d has a default value of 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 first valued was overwritten with 1 and m was overwritten to list[‘x’] and n was overwritten to value of {‘a’:’y’} because the key a has value of y in the scope