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:Output will be 1,2,8

The arguments in the funcion definiton are called as the Default arguments

If at all if we dont pass any value thye consider the default value,if we pass some value they conisder that

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:Result of above code is 1,2,3

While making a function call ,we have passed the values along with argument name

so the order does not matter because the Python interpreter is able to use the keywords provided to match the values with parameters

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

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

print(a, pargs)

>>> func(1, 2, 3)

Ans:Output will be as 1,(2,3)

***pargs is a variable length argument .They hold alll non keyword arguments .All the values passed as part of this are stored in tuple**

4. What does this code print, and why?

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

print(a, kargs)

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

Ans:

Output will be 1 {'c': 3, 'b': 2}

****kargs is called keyword varibale length arguments .We use them when we store the values in key value pair**

They return the value in form of tuple

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:

Output will be 1 5 6 5

It will not throw any errro because *(5,6) will automatically considers the values for b and c asd we did not speciify which value is for which .And for D we already have a default value

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:

Out put will be as (1, ['x'], {'a': 'y'})

This is as same as the keyword argumrnst concpet

And the above statmnets even though written in single will not throw any error as they are seprated by semicolon