**Yashwant Desai – Assignment 23**

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

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

print(a, b, c)

>>> func(1, 2)

Answer: if we fix an indentation issue the result of above code is 1 2 8 . The func function takes three parameters: a, b, and c. When the function is called with the arguments func(1, 2), the value of a is set to 1, and the value of b is set to 2. However since no value is provided for the parameter c it takes its default value which is 8.A

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)

Answer: if we fix an indentation issue the result of above code is 1 2 3 . the function is called with the arguments func(1, c=3, b=2), it explicitly assigns the value 1 to parameter a, 2 to parameter b, and 3 to parameter c hence we are getting 1 2 3 outcome.

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

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

print(a, pargs)

>>> func(1, 2, 3)

Answer: If we fix an indentation issue the result of above code is 1 (2, 3). In the func function definition, a is a regular parameter, and \*pargs is a special syntax that allows the function to accept a variable number of positional arguments. the arguments func(1, 2, 3), the value 1 is assigned to parameter a, and the rest of the positional arguments 2 and 3 are collected into a tuple due to the \*pargs parameter. This means that pargs will be a tuple containing all the positional arguments after the first one.The code will print the value of a as 1 and the value of pargs as (2, 3) since the tuple (2, 3) contains the additional positional arguments passed to the function.

4. What does this code print, and why?

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

print(a, kargs)

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

Answer: If we fix an indentation issue the result of above code is 1 {'c': 3, 'b': 2}. the value 1 is assigned to parameter a, and the rest of the keyword arguments c=3 and b=2 are collected into a dictionary due to the \*\*kargs parameter. the value of a is 1 and the value of kargs is {'c': 3, 'b': 2}

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))

Answer: A result of above code is 1 5 6 5. The function is called with the arguments func(1, \*(5, 6)), the \*(5, 6) syntax is used to unpack the tuple (5, 6) and pass its elements as positional arguments to the function. This means that 1 is assigned to parameter a, 5 is assigned to parameter b, and 6 is assigned to parameter c. Since there is no value provided for d, it takes its default value of 5 hence values of a, b, c, and d are 1, 5, 6, and 5, respectively.

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

Answer: A result of above code is (1, ['x'], {'a': 'y'}). The function func(l, m, n) and pass the variables l, m, and n as arguments to the function. b[0] = 'x' , c['a'] = 'y'. the values of l, m, and n. The values of l and n remain unchanged because they are immutable types. The value of m has changed due to the modification inside the function, and its first element has been updated to 'x' hence the final output will be (1, ['x'], {'a': 'y'})

**Regards,**

**Yashwant**