**Assignment\_23**

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

In [1]:

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

print(a, b, c)

In [2]:

func(1, 2) 1 2 8

Here as we have hardcoded and sent 2 values therefore preference is given to the input user gives. so a=1,b=2,c as by default argument 8.

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

In [3]:

**def** func(a, b, c**=**5): print(a, b, c) func(1, c**=**3, b**=**2)

1 2 3 here a=1,b=2(hardcoded by user),c=3(hardcoded by user). position doesnt matter as long as we have mentioned the variable to which value is assigned.

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

In [4]:

**def** func(a, **\***pargs): print(a, pargs) func(1, 2, 3) 1 (2, 3)

Here a=1. \*pargs returns as many argument user gives as input in form of tuples

**4. What does this code print, and why?**

In [5]:

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

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

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

Here a=1. \*\*kargs returns as many argument user gives as input in form of dictionary

1. **What gets printed by this, and explain?**

In [6]:

**def** func(a, b, c**=**8, d**=**5):

print(a, b, c, d) func(1, **\***(5, 6)) 1 5 6 5

Here a=1. User input is given in the form of **kargs. So as many variable is present that many times the value passed as** kargs input will get printed.So here a=1,b=5,c=6,d=again 5.

1. **what is the result of this, and explain?**

In [8]:

**def** func(a, b, c):

a **=** 2; b[0] **=** 'x'; c['a'] **=** 'y'

In [9]:

l**=**1; m**=**[1]; n**=**{'a':0}

In [10]:

func(l, m, n)

In [11]:

l, m, n

Out[11]:

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

Here in func(l,m,n) we passed l=1(user input), m is passed as list contain 1 element i.e 'x', n is passed as dictionary i.e {'a':'y'}