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 : 1 2 8 (Normal argument: matched by position.)**

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 : 1 2 3 (Normal argument: matched by position; Keyword argument: matched by name)**

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

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

print(a, pargs)

>>> func(1, 2, 3)

**Ans : 1 (2, 3) (Matches and collects remaining positional arguments in a tuple. Normal argument: matched by position)**

4. What does this code print, and why?

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

print(a, kargs)

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

**Ans : 1 {'c': 3, 'b': 2} (Matches and collects remaining keyword arguments in a dictionary. Keyword argument: matched by name)**

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 : 1 5 6 5 (Normal argument: matched by position. Pass all object in sequence as individual positional arguments)**

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 : (1, ['x'], {'a': 'y'}) (Pass all object in sequence as individual positional arguments)**