In [ ]:
```
#Q1
```

In [8]:
```
def func(a, b=6, c=8):
    print(a, b, c)
func(1, 2)

#In Python, when defining a function, you can assign default values to the para
#If a value is not provided for a parameter when the function is called, it wil
```

```
1 2 8
```

In [7]:
```
#Q2
```

In [9]:
```
def func(a, b, c=5):
    print(a, b, c)
func(1, c=3, b=2)
#The function func has three parameters: a, b, and c. The default value for c i
#When you call the function func(1, c=3, b=2), you are passing values for a, b,
#The values 1, 2, and 3 are assigned to a, b, and c, respectively.
```

```
1 2 3
```

In [10]:
```
#Q3
```

In [11]:
```
def func(a, *pargs):
    print(a, pargs)
func(1, 2, 3)
#The function func has a parameter a, followed by *pargs.
#The asterisk (*) before pargs indicates that it will receive any number of pos
#When you call the function func(1, 2, 3), the value 1 is assigned to a, and th
```

```
1 (2, 3)
```

In [12]:
```
#Q4
```

In [13]:
```
def func(a, **kargs):
    print(a, kargs)
func(a=1, c=3, b=2)
#When you call the function func(a=1, c=3, b=2), the value 1 is assigned to a,
#the remaining keyword arguments c=3 and b=2 are packed into the kargs dictiona
```

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

In [14]:
```
#Q5
```

In [15]:

```
def func(a, b, c=8, d=5):
    print(a, b, c, d)
func(1, *(5, 6))
#When you call the function func(1, *(5, 6)), you are passing values for a and
# and you are using the asterisk (*) to unpack the tuple (5, 6) and pass its el
```

```
1 5 6 5
```

In [16]:

In [ ]: