***Q1. What is the concept of a metaclass?***

***Ans.***A metaclass in Python is a class of a class that defines how a class behaves. A class is itself an instance of a metaclass. A class in Python defines how the instance of the class will behave.

***Q2. What is the best way to declare a class's metaclass?***

***Ans.***In Python, we can customize the class creation process by passing the metaclass keyword in the class definition. This can also be done by inheriting a class that has already passed in this keyword.

#### ***Q3. How do class decorators overlap with metaclasses for handling classes?***

***Ans***: Anything you can do with a class decorator, you can of course do with a custom metaclasses (just apply the functionality of the "decorator function", i.e., the one that takes a class object and modifies it, in the course of the metaclass's \_\_new\_\_ or \_\_init\_\_ that make the class object!).

#### ***Q4. How do class decorators overlap with metaclasses for handling instances?***

***Ans:*** Anything you can do with a class decorator, you can of course do with a custom metaclass (just apply the functionality of the "decorator function", i.e., the one that takes a class object and modifies it, in the course of the metaclass's \_\_new\_\_ or \_\_init\_\_ that make the class object!).