Q1. What is the concept of a metaclass?

It’s a class of a class which defines how a class should behave. Class itself is an instance of a metaclass.

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

In the below manner we can declare the class’s metaclass explicitly, where we are inheriting the type class.

Class Meta(type):

Pass

Class NewMeta(metaclass = Meta):

pass

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

Metaclasses can’t be composed together easily, while many class decorators can be used to extend the same class without conflicts.

Decorators are useful when we need to modify the class methods, class attributes.

Whenever we use a class, for example to instantiate it, first Python *builds* that class using the metaclass and the class definition we wrote

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

Same question as Q3.