ASSIGNMENT - 11

Q1. What is the concept of a metaclass?

Ans: A metaclass is a class of a class. In Python, everything is an object, including classes. Metaclasses allow you to define how classes themselves behave. They define the creation and behavior of classes at a higher level.

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

Ans: The most common way to declare a class's metaclass is by using the metaclass keyword argument in the class definition.

For example:

class MyMeta(type):

# Metaclass definition

class MyClass(metaclass=MyMeta):

# Class definition

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

Ans:

* Class decorators and metaclasses both provide ways to modify or enhance class behavior during or after class creation.
* Class decorators work on the class object directly after it's created, while metaclasses intervene at the moment the class itself is created.
* Both can be used to add functionality, modify attributes, or perform operations on the class being defined.
* Class decorators are more straightforward and can be applied more selectively to individual classes, while metaclasses affect the behavior of entire hierarchies of classes.

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

Ans:

* Class decorators primarily affect class-level behavior, while metaclasses deal with the creation and behavior of classes.
* Metaclasses can influence instance creation by modifying the class's \_\_new\_\_ or \_\_init\_\_ methods, allowing control over how instances are instantiated.
* Class decorators, however, do not directly interact with instances; their focus is on modifying or enhancing the class itself.