ADVANCE ASSIGNMENT - 2

Q1. What is the relationship between classes and modules?

Ans1- classes and modules are complementary concepts that work together to help structure and organize code in object-oriented programming. Modules provide a way to organize and manage classes, functions, and variables, while classes define the blueprint for creating objects with specific attributes and behaviors.

Q2. How do you make instances and classes?

Ans2-a)creating class:

class MyClass:

pass

b)creating instance:

obj = MyClass()

Q3. Where and how should be class attributes created?

Ans3- Class attributes should be defined at the class level, outside of any instance methods, directly within the class definition.

Q4. Where and how are instance attributes created?

Ans4- Instance attributes are typically created and initialized within the constructor (\_\_init\_\_) method of a class in Python. They are specific to each instance and are assigned values based on constructor arguments during object creation.

Q5. What does the term "self" in a Python class mean?

Ans5- the term "self" refers to the instance of the class itself. It is a convention, not a keyword, and is used as the first parameter in instance method definitions, including the constructor (\_\_init\_\_). It represents the specific instance on which the method is called, allowing you to access and manipulate its attributes and behaviors

Q6. How does a Python class handle operator overloading?

Ans6- operator overloading is achieved by defining special methods in a class with double underscores (e.g., \_\_add\_\_, \_\_sub\_\_) that specify how instances of the class should behave when operated with standard operators like +, -, \*, etc. These special methods allow you to customize the behavior of operators for instances of your class.

Q7. When do you consider allowing operator overloading of your classes?

Ans7- You should consider allowing operator overloading for your classes when it makes the code more intuitive, readable, and follows common conventions for the operations you're overloading. Use it when it enhances the usability and expressiveness of your class.

Q8. What is the most popular form of operator overloading?

Ans8- The most popular form of operator overloading in Python is overloading the addition operator + using the \_\_add\_\_ method to define custom behavior for adding instances of a class.

Q9. What are the two most important concepts to grasp in order to comprehend Python OOP code?

Ans9- The two most important concepts to grasp in order to comprehend Python OOP code are:

Classes and Objects: Understand how classes define blueprints for objects and how objects encapsulate both data (attributes) and behavior (methods).

Inheritance and Polymorphism: Learn how inheritance allows you to create new classes based on existing ones and how polymorphism enables objects of different classes to be treated as objects of a common superclass.