ASSIGNMENT - 2

Q1. What is the relationship between classes and modules?

Ans: In Python, modules are files containing Python code, while classes are blueprints for creating objects. Classes can be defined within modules, allowing the module to contain various classes alongside other functions and variables. Modules serve as containers for classes and provide a way to organize and reuse code.

Q2. How do you make instances and classes?

Ans: To create instances of a class, you use the class name followed by parentheses. For example:

* Creating a class: class MyClass:
* Creating an instance: obj = MyClass()

Q3. Where and how should be class attributes created?

Ans: Class attributes are defined within the class block, outside of any class methods. They are shared by all instances of the class and can be accessed using the class name. Class attributes are often initialized directly within the class scope.

Q4. Where and how are instance attributes created?

Ans: Instance attributes are created within the \_\_init\_\_ method of a class. They are specific to each instance of the class and are initialized using the self keyword within the \_\_init\_\_ method.

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

Ans: In Python, self refers to the instance of the class itself. It's the first parameter of instance methods and is used to access and manipulate instance attributes and methods within the class.

Q6. How does a Python class handle operator overloading?

Ans: Python classes can define special methods (like \_\_add\_\_, \_\_sub\_\_, etc.) that allow instances of the class to use built-in operators (+, -, etc.) in a customized way. This is called operator overloading, where the behavior of operators is defined for specific classes.

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

Ans: Operator overloading is useful when you want instances of your class to behave naturally with Python's built-in operators. For example, defining addition (\_\_add\_\_) for a custom class allows instances of that class to use the + operator.

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

Ans: Overloading the addition operator (\_\_add\_\_) is quite common because it allows objects of a class to be added using the + operator, providing intuitive behavior for users.

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

Ans: The two most crucial concepts to understand in Python OOP are:

* Class and Instance: Understanding how classes define the behavior and structure of objects, while instances are specific realizations of those classes with their own attributes and behaviors.
* Inheritance and Polymorphism: grasping how classes can inherit attributes and behaviors from other classes, allowing for code reuse and polymorphic behavior where different objects can be treated as instances of a common superclass.