**Python advance assignment-2**

**Q1. What is the relationship between classes and modules**?

A1. Classes and modules are related in that a module can contain one or more classes. A class is a blueprint for creating objects (instances), while a module is a file containing Python definitions and statements.

**Q2. How do you make instances and classes?**

A2. To make instances of a class, you use the class name followed by parentheses, like so: my\_instance = MyClass(). To create a class, use the class keyword, like so: class MyClass:

**Q3. Where and how should be class attributes created?**

A3. Class attributes are created within the class definition and are shared among all instances of the class. They should be defined at the top level of the class, outside of any method definitions.

**Q4. Where and how are instance attributes created?**

A4. Instance attributes are created within the **init** method of a class and are specific to each instance of the class. They should be defined within the **init** method and assigned to self.

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

A5. The term "self" in a Python class refers to the instance of the class. It is used to access instance attributes and methods.

**Q6. How does a Python class handle operator overloading?**

A6. Python classes can handle operator overloading by defining special methods that correspond to specific operators. For example, the + operator is overloaded by defining the **add** method.

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

A7. You should consider allowing operator overloading in your classes if it makes sense in the context of the class and it improves the readability and usability of the class.

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

A8. The most popular form of operator overloading in Python is using special methods to overload the built-in operators.

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

A9. The two most important concepts to grasp in order to comprehend Python OOP code are classes and objects. Understanding how classes are used to create objects and how objects interact with each other is crucial for understanding OOP in Python.