Q1. What is the relationship between classes and modules?

Answer> classes are blueprints of objects which contain members like fields and methods. Module introduces the definition related to properties, events, variables and procedures of its members. Modules cannot be instantiated like classes.

Q2. How do you make instances and classes?

Answer> The *class* statement creates a new class definition. The name of the class immediately follows the keyword *class* followed by a colon

Class ClassName:

Statement

To create instances of a class, call the class using class name and pass in whatever arguments its *\_\_init\_\_* method accepts

"This would create first object of Employee class"

emp1 = Employee("Zara", 2000)

Q3. Where and how should be class attributes created?

Answer> Class attributes belong to the class itself they will be shared by all the instances. Such attributes are defined in the class body parts usually at the top, for legibility. Unlike class attributes, instance attributes are not shared by objects.

Q4. Where and how are instance attributes created?

Answer> Instance Attributes are unique to each object, (an instance is another name for an object)

An instance attribute is a Python variable belonging to one, and only one, object. This variable is only accessible in the scope of this object and it is defined inside the constructor function, \_\_init\_\_(self,..) of the class

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

Answer> The self parameter is a reference to the current instance of the class, and is used to access variables that belongs to the class.

Q6. How does a Python class handle operator overloading?

Answer> The operator overloading in Python means provide extended meaning beyond their predefined operational meaning. Such as, we use the "+" operator for adding two integers as well as joining two strings or merging two lists. We can achieve this as the "+" operator is overloaded by the "int" class and "str" class.

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

Answer> we have two objects which are a physical representation of a class (user-defined data type) and we have to add two objects with binary '+' operator it throws an error, because compiler don't know how to add two objects. So we define a method for an operator and that process is called operator overloading.

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

Answer> A very popular and convenient example is the Addition (+) operator. Just think how the '+' operator operates on two numbers and the same operator operates on two strings. It performs “Addition” on numbers whereas it performs “Concatenation” on strings.

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

Answer> Inheritance and polymorphism are fundamental concepts of object oriented programming. These concepts help us to create code that can be extended and easily maintainable