Q1. What is the relationship between classes and modules?

Modules are files that can contain Python code, including class definitions. Classes defined in a module can be imported and used in other parts of your code, allowing for a modular and organized approach to programming.

Q2. How do you make instances and classes?

We define a class using the class keyword.

Create an instance by calling the class like a function, which invokes the \_\_init\_\_ method.

Q3. Where and how should be class attributes created?

Class attributes in Python are defined within a class but outside of any methods. They are shared across all instances of the class and can be accessed using the class name or an instance of the class.

Q4. Where and how are instance attributes created?

Instance attributes in Python are created and initialized within the \_\_init\_\_ method of a class. These attributes are specific to each instance of the class, meaning that each instance can have different values for these attributes.

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

the term self refers to the instance of the class on which a method is called. It is a conventional name used for the first parameter of instance methods, though you could technically use any name.

Q6. How does a Python class handle operator overloading?

In Python, operator overloading allows you to define custom behavior for operators (like +, -, \*, etc.) when applied to instances of your classes. You achieve this by implementing special methods in your class, also known as magic methods or dunder methods (short for double underscore).

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

Allow Operator Overloading when it makes operations on class instances intuitive and aligns with how built-in types behave.

Ensure Consistency with standard operators to maintain readability and usability.

Use Judiciously to avoid confusion and maintain code clarity.

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

The most popular forms of operator overloading in Python typically involve arithmetic and comparison operators. These operators are commonly overloaded because they are fundamental to many data types and operations

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

Classes and Instances: Grasp how classes are used to create instances and how instances interact with attributes and methods.

Inheritance: Understand how inheritance allows classes to reuse and extend functionality, establishing relationships and hierarchies between classes.