Q1. What is the relationship between classes and modules?

Classes is a way to define objects with properties and functions, while modules are files containing python code. module can contain class definitions, helps us to organize related classes and reusing them in different programs, making it easier to manage and maintain code.

Q2. How do you make instances and classes?

First we have to define a class by using the class keyword, with all its properties and functions and the

create instances of the class by calling the class name followed by parentheses , which invokes the class constructor, which initializes the instance.

Q3. Where and how should be class attributes created?

Class attributes are created inside the class definition, outside of methods, and can be accessed using the class name or instances.

Q4. Where and how are instance attributes created?

Instance attributes are created inside the methods of a class using the self keyword. Each object of the class can have its own unique attributes.

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

"self" refers to the instance of the class itself. It is used to represent the object on which methods are invoked, allow us to access to its attributes and function.

Q6. How does a Python class handle operator overloading?

A python class can change how operators like + or > work for its instances by defining special methods like \_\_add\_\_ .\_\_gt\_\_. It allows us to customize behavior when using operators with objects of that class.

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

when it makes sense in the context of problem. Like, if your class represents a game character, overloading operators like + or > can make it easier to compare or combine characters in the game.

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

Arithmetic operator overloading, which involves defining special methods like \_\_add\_\_, \_\_sub\_\_,etc., to customize the behavior of arithmetic operations on class objects.

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

classes and objects. Understanding how classes define blueprints for objects and how objects are created from classes.