Q1. What is the relationship between classes and modules?

Classes may generate instances (objects), and have per-instance state (instance variables). Modules may be mixed in to classes and other modules. The mixed in module's constants and methods blend into that class's own, augmenting the class's functionality. Classes, however, cannot be mixed in to anything.

Q2. How do you make instances and classes?

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

Q3. Where and how should be class attributes created?

A class attribute is shared by all instances of the class. To define a class attribute, you place it outside of the \_\_init\_\_() method. Use class\_name. class\_attribute or object\_name.

Q4. Where and how are instance attributes created?

Instance attributes are defined in the constructor. Defined directly inside a class. Defined inside a constructor using the self parameter.

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

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?

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?

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.

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

if the user wants to make the operator “+” to add two class objects, the user has to redefine the meaning of the “+” operator such that it adds two class objects. This is done by using the concept “Operator overloading”.

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

In this article, we will elaborate on two key concepts of OOP which are inheritance and polymorphism. Both inheritance and polymorphism are key ingredients for designing robust, flexible, and easy-to-maintain software. These concepts are best explained via examples. Let's start with creating a simple class.