Q1. What is the relationship between classes and modules?

**Ans: Modules are collections of methods and constants. They cannot generate instances. Classes may generate instances (objects), and have per-instance state (instance variables).**

Q2. How do you make instances and classes?

**Ans: 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?

**Ans: Class attributes are attributes which are owned by the class itself. They will be shared by all the instances of the class.**

Q4. Where and how are instance attributes created?

**Ans: 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?

**Ans: Self represents the instance of the class. By using the “self” keyword we can access the attributes and methods of the class in python.**

Q6. How does a Python class handle operator overloading?

**Ans: Python operators work for built-in classes. For example, the + operator will perform arithmetic addition on two numbers, merge two lists, or concatenate two strings. This feature in Python that allows the same operator to have different meaning according to the context is called operator overloading.**

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

**Ans: Operator Overloading means giving extended meaning beyond their predefined operational meaning. For example operator + is used to add two integers as well as join two strings and merge two lists. It is achievable because '+' operator is overloaded by int class and str class**

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

**Ans : the + operator will perform arithmetic addition on two numbers, merge two lists, or concatenate two strings. This feature in Python that allows the same operator to have different meaning according to the context is called operator overloading.**

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

**Ans : In python Object-oriented programming we have four important concept Inheritance, Encapsulation, Polymorphism, and abstraction. It is very important to know about all of these in order to understand OOPs.**