Q1. What is the relationship between classes and modules?

Ans:

The difference between a class and a module in python is that a class is used to define a blueprint for a given object, whereas a module is used to reuse a given piece of code inside another program.

Q2. How do you make instances and classes?

Ans:

Instance is an object that belongs to a class. For instance, list is a class in Python. When we create a list, we have an instance of the list class. Class variables are declared inside a class but outside of any function. Instance variables are declared inside the constructor which is the \_\_init\_\_method.

Q3. Where and how should be class attributes created?

Ans:

Class attributes are the variables defined directly in the class that are shared by all objects of the class.

Q4. Where and how are instance attributes created?

Ans:

Instance attributes are attributes or properties attached to an instance of a class. Instance attributes are defined in the constructor

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. It binds the attributes with the given arguments.  
The reason you need to use self. is because Python does not use the @ syntax to refer to instance attributes. Python decided to do methods in a way that makes the instance to which the method belongs be passed automatically, but not received automatically: the first parameter of methods is the instance the method is called on.

Q6. How does a Python class handle operator overloading?

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. You might have noticed that the same built-in operator or function shows different behavior for objects of different classes, this is called Operator Overloading. When we use an operator on user defined data types then automatically a special function or magic function associated with that operator is invoked. Changing the behavior of operator is as simple as changing the behavior of method or function. You define methods in your class and operators work according to that behavior defined in methods. When we use + operator, the magic method \_\_add\_\_ is automatically invoked in which the operation for + operator is defined. There by changing this magic method’s code, we can give extra meaning to the + operator.

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

Ans:

Operator overloading is the process of using an operator in different ways depending on the operands. You can change the way an operator in Python works on different data-types.

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

Ans:

 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?

Ans:

inheritance and polymorphism.