Q1. What is the relationship between classes and modules?

Modules are collections of methods and constants. They cannot generate instances. 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?

use the keyword class to create class & 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?

Class Attributes are unique to each class. Each instance of the class will have attribute. Which is created inside the class.

Q4. Where and how are instance attributes created?

Instance Attributes are unique to each object, (an instance is another name for an object).Which is created inside the initializer.

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

self represents the instance of the class.

Q6. How does a Python class handle operator overloading?

To achieve operator overloading, we define a special method in a class definition. The name of the method should begin and end with a double underscore (\_\_). The + operator is overloaded using a special method named \_\_add\_\_(). This method is implemented by both the int and str classes.

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

Overloading operators is useful with classes because there are times when you want to use a class object with an operator that does not recognize that object

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

The most frequent instance is the adding up operator ‘+’, where it can be used for usual addition and also for combining two different strings.

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

inheritance and polymorphism