***Q1. What is the relationship between classes and modules?***

***A module in python is simply a way to organize the code, and it contains either python classes or just functions. If we need those classes or functions,just import them.***

***Q2. How do you make instances and classes?***

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

***create a class, using the keyword class***

***Create an Object and the self Parameter***

***Q3. Where and how should class attributes be created?***

***Class attributes are attributes which are owned by the class itself. They will be shared by all the instances of the class. Therefore they have the same value for every instance. We define class attributes outside all the methods, usually they are placed at the top, right below the class header.***

***Q4. Where and how are instance attributes created?***

***An instance attribute is a Python variable belonging to only one object. It is only accessible in the scope of the object and it is defined inside the constructor function of a class.***

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

***Q6. How does a Python class handle operator overloading?***

***To perform operator overloading, Python provides some special function that is automatically invoked when it is associated with that particular operator.***

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

***It allows you to provide an intuitive interface to users of your class, plus makes it possible for templates to work equally well with classes and built-in/intrinsic types***

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

***A very popular and convenient is the Addition (+) operator.***

***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?***

***Inheritance, Encapsulation, Polymorphism, and Data abstraction***