Q1. What is the relationship between classes and modules?

An object is simply a collection of data (variables) and methods (functions) that act on those data.

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

Classes may generate instances (objects), and have pre-instance state (instance variables)

Q2. How do you make instances and classes?

Class-> To create a class, use the keyword class

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

Q3. Where and how should be class attributes 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 instances 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. For example:- \_\_init\_\_(self..,)

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

“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?

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?

[Python operators](https://www.programiz.com/python-programming/operators) work for built-in classes. But the same operator behaves differently with different types. This feature in Python that allows the same operator to have different meaning according to the context is called operator overloading. It does not limit operator overloading to arithmetic operators only. We can overload comparison operators as well.

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

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?

Both inheritance and polymorphism are key ingredients for designing robust, flexible, and easy-to-maintain software.