Q1. What is the relationship between classes and modules?

A:

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?

A: class Dog:

# A simple class

# attribute

attr1 = "mammal"

attr2 = "dog"

# A sample method

def fun(self):

print("I'm a", self.attr1)

print("I'm a", self.attr2)

# Driver code

# Object instantiation

Rodger = Dog()

# Accessing class attributes

# and method through objects

print(Rodger.attr1)

Rodger.fun()

Q3. Where and how should be class attributes created?

A:

class Circle:

def \_\_init\_\_(self, radius):

self.pi = 3.14159

self.radius = radius

def area(self):

return self.pi \* self.radius\*\*2

def circumference(self):

return 2\*self.pi \* self.radius

Q4. Where and how are instance attributes created?

A:

Instance attributes are defined in the constructor. Defined directly inside a class. Defined inside a constructor using the self parameter.

Q5. What does the term &quot;self&quot; in a Python class mean?

A:

self represents the instance of the class. By using the “self”  we can access the attributes and methods of the class in python. It binds the attributes with the given arguments.

Q6. How does a Python class handle operator overloading?

A:

class operator:

def \_\_init\_\_(self, a, b):

self.a = a

self.b = b

def func1(self):

return self.a+self.b

a=operator(2,3)

b=operator(['A', 'B'] , [1, 2, 3])

print(a.func1())

print(b.func1())

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

A:

When one or both operands are of a user-defined class or structure type, operator overloading makes it easier to specify user-defined implementation for such operations. This makes user-defined types more like the basic primitive data types in terms of behaviour.

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

A:

A 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

A:

Both inheritance and polymorphism are fundamental concepts of object oriented programming. These concepts help us to create code that can be extended and easily maintainable.