Q1. What is the relationship between classes and modules?

Ans:

Class is like blue print of identical items being identical to each other in various aspect vs module are collection of class , function , variables under same roof .

Q2. How do you make instances and classes?

Ans:

Classes are simply made by calling class and defining function or either constructor to call the instance within it . Instances are made by by calling the class for which instance is to create and passing the arguments to the variables .

Ex :

*# Class car with details of body,engine,tyre*

class car :  
 def \_\_init\_\_(self,body,engine,tyre):  
 self.body1=body  
 self.engine1=engine  
 self.tyre1=tyre  
  
*# Instance of the class car*   
c= car('SOlid','V8','Alloy wheel')

Q3. Where and how should be class attributes created?

Ans:

Class attributes be created within the class which are shared to all the class instances.

They are usually created at the top after defining the class.

Ex:

class car :  
 *# class attribute* count =0  
  
 def car\_popu(self):  
 car.count += 1  
  
chiron = car()  
chiron.car\_popu()  
print(chiron.count)

Q4. Where and how are instance attributes created?

Ans:

It’s the class variable belonging to only one object .It’s is usually defined inside constructor

Function of class.

Ex:

class car :  
 def \_\_init\_\_(self,body,engine,tyre):

# all below variable are instance attributes for object car1.  
 self.body1=body  
 self.engine1=engine  
 self.tyre1=tyre

Car1 = car('SOlid','V8','Alloy wheel')

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

Ans:

Self represent the instances of the class and is used to bind the attributes with the argument .By using self we can access the attributes and methods of the class.

Q6. How does a Python class handle operator overloading?

Ans:

Python class handle the operator overloading by identifying the context the data and performing the operation accordingly .

Ex:

Same add function add str , integer . float values

Thus python identify the context of data and perform accordingly.

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

Ans:

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

Ans:

Add is most popular form of operator overloading

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

Ans:

Inheritance and Polymorphism .