**Q1. Define the relationship between a class and its instances. Is it a one-to-one or a one-to-many partnership, for example?**

Answer : Relationship between a class and its instances is a one to many partnership.

**Q2. What kind of data is held only in an instance?**

Answer : I*nstance objects contains the Instance variables which are specific to that specific Instance object.*

**Q3. What kind of knowledge is stored in a class?**

Answer : *Class creates a user-defined data structure, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A class is like a blueprint for an object.*

**Q4. What exactly is a method, and how is it different from a regular function?**

Answer : *The methods with a class can be used to access the insatnce variables of its instance.So, the object's state can be modified by its method.Function cant access the attributes of an instance of a class or cant modify the state of the object.*

**Q5. Is inheritance supported in Python, and if so, what is the syntax?**

Answer : Python supports inheritance.

class Person:  
  def \_\_init\_\_(self, fname, lname):  
    self.firstname = fname  
    self.lastname = lname

class Student(Person):  
  pass

**Q6. How much encapsulation (making instance or class variables private) does Python support?**

Answer : Encapsulation describes the idea of wrapping data and the methods that work on data within one unit. This puts restrictions on accessing variables and methods directly and can prevent the accidental modification of data. To prevent accidental change, an object’s variable can only be changed by an object’s method.

**Q7. How do you distinguish between a class variable and an instance variable?**

Answer :*The class attribute is available to all the instance objects of that class.*

*Instance variable is accessible only to the object or Instance of that class*

**Q8. When, if ever, can self be included in a class's method definitions?**

Answer : *self can included to access the class variables and instance*

*variables.*

**Q9. What is the difference between the \_ \_add\_ \_ and the \_ \_radd\_ \_ methods**?

Answer : Entering \_\_radd\_\_. Python will first try \_\_add\_\_(), and if that returns NotImplemented Python will check if the right-hand operand implements **radd**, and if it does, it will call \_\_radd\_\_() rather than raising a TypeError.

**Q10. When is it necessary to use a reflection method? When do you not need it, even though you support the operation in question?**

Answer : Reflection method we often encounter the requirement that a method in the executing object, or a variable in the calling object, or a field of the object should be assigned, while the method name or field name can not be determined when encoding the code, and need to be input in the form of passing strings through parameters.

**Q11. What is the \_ \_iadd\_ \_ method called?**

Answer : *\_\_iadd\_\_ method is called when we use implementation like a+=b which is a.\_\_iadd\_\_(b)*