**Q1. Define the relationship between a class and its instances. Is it a one-to-one or a one-to-many**

**partnership, for example?**

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

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

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

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

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

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

**Q8. When, if ever, can self be included in a class&#39;s method definitions?**

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

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

**support the operation in question?**

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

**Q12. Is the \_ \_init\_ \_ method inherited by subclasses? What do you do if you need to customize its**

**behavior within a subclass?**

**SOLUTIONS**

*1. A class is a blueprint or template for creating objects, while instances are the actual objects created based on the class. The relationship between a class and its instances is a one-to-many partnership, as a single class can have multiple instances created based on it.*

*2. Instance data is held only in an instance, which includes data unique to that particular instance. This can include instance variables that are set and manipulated within the instance's methods.*

*3. A class stores knowledge that applies to all instances of that class. This includes class variables, which are shared among all instances of the class, as well as methods that define the behaviors and actions of the class.*

*4. A method is a function that is defined within a class and is designed to operate on instances of that class. The main difference between a method and a regular function is that a method is associated with an object or instance, while a regular function is not.*

*5. Yes, inheritance is supported in Python. The syntax for inheritance is as follows:*

***class SubClass(ParentClass):***

***# subclass definition here***

*6. Python supports encapsulation through the use of private variables, denoted by a leading underscore. However, these variables are not completely inaccessible from outside the class.*

*7. A class variable is shared among all instances of a class, while an instance variable is unique to a particular instance. Class variables are defined within the class definition but outside of any method, while instance variables are usually defined within a class method.*

*8. The self parameter is included in all method definitions for a class, as it is used to refer to the instance that the method is operating on.*

*9. The* ***add*** *method is used to define the behavior of the + operator when applied to instances of a class, while the* ***radd*** *method is used when the + operator is applied to an instance of another class that does not support addition with the original class.*

*10. Reflection is necessary when you need to dynamically access and modify an object's attributes at runtime, based on their names. Reflection can be avoided when you already know the names and types of the attributes you need to access or modify, as you can directly use them in your code.*

*Q11. The* ***iadd*** *method is called the in-place addition method, as it is used to define the behavior of the += operator when applied to instances of a class.*

*Q12. The* ***init*** *method is inherited by subclasses, but can be customized within a subclass by defining a new* ***init*** *method with different behavior, or by calling the parent class's* ***init*** *method using the super() function.*