Q1. The relationship between a class and its instances is a one-to-many partnership. A class serves as a blueprint or template to create multiple instances (objects) with the same characteristics and behaviors.

Q2. Instance variables hold data that is unique to each instance of a class.

Q3. A class stores knowledge in the form of class variables and methods that describe the behavior of the class and its instances.

Q4. A method is a function defined inside a class and is used to perform an operation on an instance of that class. A method is different from a regular function because it is called on an instance of a class and has access to the instance's data.

Q5. Yes, inheritance is supported in Python, and it is declared using the syntax class Child(Parent): where Child is the derived class, and Parent is the base class.

Q6. Python supports encapsulation by allowing instance variables and methods to be marked as private by prefixing their names with two underscores (e.g., \_\_variable). However, Python does not provide true encapsulation as these attributes can still be accessed from outside the class.

Q7. A class variable is shared by all instances of a class, whereas an instance variable is unique to each instance. A class variable is defined inside the class definition, but outside any method, and is accessed using the class name. An instance variable is defined inside a method using the self keyword and is accessed using the dot notation on an instance of the class.

Q8. The self parameter is included in a class's method definitions to reference the instance that the method is being called on.

Q9. The \_\_add\_\_ method is used to implement the "+" operator, while the \_\_radd\_\_ method is used to implement the "+" operator when the left-hand operand does not support it.

Q10. Reflection methods are necessary when you need to access or modify an object's attributes or methods dynamically at runtime. You do not need reflection if you know the name of the attribute or method at the time of writing the code.

Q11. The \_\_iadd\_\_ method is called when the "+=" operator is used on an instance of a class.

Q12. The \_\_init\_\_ method is inherited by subclasses, and it can be customized within a subclass by defining a new \_\_init\_\_ method with the desired behavior. To call the parent class's \_\_init\_\_ method from within a subclass's \_\_init\_\_ method, use the super() function.