Q1. What is the meaning of multiple inheritance?

Multiple inheritance refers to a situation in Python where a class inherits from multiple parent classes. This is in contrast to single inheritance, where a class inherits from only one parent class.

Multiple inheritance can be useful in situations where you want to reuse functionality from multiple classes. For example, you might have a class that represents a car, and you might also have a class that represents a flying object. You could then create a class that inherits from both of these classes, and this new class would have all of the functionality of both classes.

Q2. What is the concept of delegation?

Delegation is a design pattern in which an object, called the delegate, is responsible for performing certain tasks on behalf of another object, called the delegator. This can be done by the delegator forwarding method calls and attribute access to the delegate.

Q3. What is the concept of composition?

What's Composition? Composition is a concept that models a has a relationship. It enables creating complex types by combining objects of other types. This means that a class Composite can contain an object of another class Component . This relationship means that a Composite has a Component .

Q4. What are bound methods and how do we use them?

A bound method in Python is the method that has an object associated with it. Unbound method doesn't have an object associated with it. The concept of “unbound methods” has been removed from the language as of 3.0. When referencing a method as a class attribute, you now get a plain function object.

Q5. What is the purpose of pseudoprivate attributes?

Using pseudoprivate attributes can help in organizing your code by clearly indicating which attributes are intended for internal use within the class and which are meant to be accessed from outside the class.