Q1. What is the meaning of multiple inheritance?

Q2. What is the concept of delegation?

Q3. What is the concept of composition?

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

Q5. What is the purpose of pseudoprivate attributes?

Solutions

Q1. Multiple inheritance refers to the ability of a programming language to allow a class to inherit from multiple parent classes. In other words, a class can inherit attributes and behaviors from more than one base class. This allows the derived class to have characteristics from multiple sources, combining the features of the parent classes.

Q2. The concept of delegation involves one object passing on a specific task or responsibility to another object to perform. It is a design pattern where an object forwards a method call to another object, known as the delegate, to handle the request. Delegation helps in achieving code reuse and modularity by separating concerns and promoting loose coupling between objects.

Q3. Composition is a concept in object-oriented programming where a class is composed of one or more objects of other classes. In composition, an object is made up of other objects, and the composed objects cannot exist independently without the container object. It allows for creating complex objects by combining simpler objects, promoting code reuse, and achieving modularity.

Q4. Bound methods are methods that are associated with a specific instance of a class. When a method is bound to an instance, it is automatically passed the instance as the first argument (commonly named `self` in Python). Bound methods can access and modify the instance's attributes and invoke other methods on the instance. They are typically used to perform operations that are specific to a particular instance of a class.

To use a bound method, you call it on an instance of the class, and Python automatically passes the instance as the first argument. For example, if you have an instance `obj` of a class `MyClass` with a bound method called `my\_method`, you can invoke it as `obj.my\_method()`.

Q5. Pseudoprivate attributes, also known as name mangling, are a naming convention used in some programming languages, including Python. Pseudoprivate attributes are prefixed with double underscores (`\_\_`) but do not end with double underscores. This naming convention indicates that the attribute is intended to be private to the class where it is defined.

The purpose of pseudoprivate attributes is to provide a way to encapsulate data within a class and prevent accidental access or modification from outside the class. However, it's important to note that pseudoprivate attributes are not truly private, as they can still be accessed using a modified name. They are primarily a naming convention and a way to signal that the attribute should be treated as internal to the class.