**Q1. What is the meaning of multiple inheritance?**

Multiple inheritance is when a class is derived from more than one base class. The derived class will inherit the features of all the base classes.

**Q2. What is the concept of delegation?**

Delegation is an object oriented technique (also called a design pattern). Let's say you have an object x and want to change the behaviour of just one of its methods. You can create a new class that provides a new implementation of the method you're interested in changing and delegates all other methods to the corresponding method of x.

**Q3. What is the concept of 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. [1]

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

A bound method is the one which is dependent on the instance of the class as the first argument. It passes the instance as the first argument which is used to access the variables and functions. In Python 3 and newer versions of python, all functions in the class are by default bound methods.

**Q5. What is the purpose of pseudo private attributes?**

In Python, all instance attributes wind up in the single instance object at the bottom of the class tree. In a multi user environment where two different users tries to update an instance attribute in isolation may cause inconsistencies. To avoid this, we can use pseudo private attributes by prefixing double underscores. When thus prefixed, the attributes will be expanded to include the names of their classes before being added to the instance. As the expansion makes the names unique within the instance, the class coders can safely assume that they truly own any names that they prefix with two underscores. [2]

**References**:

1. <https://realpython.com/inheritance-composition-python/>
2. <https://www.pythonstudio.us/object-oriented/why-use-pseudoprivate-attributes.html>