Q1. What is the meaning of multiple inheritance?

**Ans. Multiple inheritance is a feature of some object-oriented computer programming languages in which an object or class can inherit characteristics and features from more than one parent object or the parent class .**

Q2. What is the concept of delegation?

**Ans. 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?

**Ans. In composition one of the classes is composed of one or more instance of other classes. In other words one class is container and other class is content and if you delete the container object then all of its contents objects is also deleted. In this concept, we will describe a class that references to one or more objects of other classes as an Instance variable. Here, by using the class name or by creating the object we can access the members of one class inside another class. It enables creating complex types by combining objects of different classes. It means that a class Composite can contain an object of another class Component.**

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

**Ans. If a function is an attribute of class and it is accessed via the instances, they are called bound methods. A bound method is one that has ‘self‘ as its first argument. Since these are dependent on the instance of classes, these are also known as instance methods. For example, if there is a class “Fruits”, and instances like apple, orange, mango are possible. Each instance may have different size, color, taste, and nutrients in it. Thus to alter any value for a specific instance, the method must have ‘self’ as an argument that allows it to alter only its property.**

Q5. What is the purpose of pseudoprivate attributes?

**Ans. The problem that the pseudo-private attribute feature is meant to alleviate has to do with the way instance attributes are stored. In Python, all instance attributes wind up in the single instance object at the bottom of the class tree, where each class gets its own space for data members it defines. Pseudo-private methods does not allow direct access to the outside world and Pseudo private property, can not be directly accessible to the outside world.**