Q1. What is the meaning of multiple inheritance?

If a child class is inheriting the properties of a single other class, we call it single inheritance. However, if a child class inherits from more than one class, i.e. this child class is derived from multiple classes, we call it multiple inheritance in Python.

Q2. What is the concept of delegation?

In object-oriented programming, delegation refers to evaluating a member (property or method) of one object (the receiver) in the context of another original object (the sender). Delegation can be done explicitly, by passing the sending object to the receiving object, which can be done in any object-oriented language; or implicitly, by the member lookup rules of the language, which requires language support for the feature. Implicit delegation is the fundamental method for behavior reuse in prototype-based programming, corresponding to inheritance in class-based programming. The best-known languages that support delegation at the language level are Self, which incorporates the notion of delegation through its notion of mutable parent slots that are used upon method lookup on self calls, and JavaScript; see JavaScript delegation.

Q3. What is the concept of composition?

Composition is one of the fundamental concepts in object-oriented programming. It describes a class that references one or more objects of other classes in instance variables. This allows you to model a has-a association between objects.

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.

class A:

def func(self, arg):

self.arg = arg

print("Value of arg = ", arg)

# Creating an instance

obj = A()

# bound method

print(obj.func)

Q5. What is the purpose of pseudoprivate attributes?

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. This is very different from the C++ model, where each class gets its own space for data members it defines.

Within a class method in Python, whenever a method assigns to a self attribute (e.g., self.attr=value), it changes or creates an attribute in the instance (inheritance search only happens on reference, not assignment). Because this is true even if multiple classes in a hierarchy assign to the same attribute, collisions are possible.