Q1. What is the meaning of multiple inheritance?

Ans: Inheritance is the mechanism to achieve the re-usability of code as one class(child class) can derive the properties of another class(parent class). It also provides transitivity ie. if class C inherits from P then all the sub-classes of C would also inherit from P. When a class is derived from more than one base class it is called multiple Inheritance. The derived class inherits all the features of the base case.

Q2. What is the concept of delegation?

Ans: 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. The delegator passes requests for certain actions to the delegate, which then performs the actions on behalf of the delegator.

Q3. What is the concept of composition?

Ans: Composition is one of the important concepts of Object-oriented programming (OOPs). Composition basically enables us for creating complex types objects by combining other types of objects in the program. Composition represents ‘has a relationship’ type or we can call composition as ‘has a relationship’ in the OOPs concept. It means that a composite class present in the program can contains the objects from the other class components and this type of relationship model is called as has a relationship.

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

Ans: 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 pseudoprivate attributes?

Ans: This trick can avoid potential name collisions in the instance, but note that it does not amount to true privacy. If you know the name of the enclosing class, you can still access either of these attributes anywhere you have a reference to the instance by using the fully expanded name (e.g., I.\_C1\_\_X = 77)