Q1. What is the meaning of multiple inheritance?

Multiple inheritance in Python refers to the ability of a class to inherit attributes and methods from more than one parent class. This allows a subclass to inherit from multiple superclasses, incorporating their functionality into a single class.

Example:

class A:

def method\_A(self):

print("Method A")

class B:

def method\_B(self):

print("Method B")

class C(A, B):

pass

obj = C()

obj.method\_A() # Method A

obj.method\_B() # Method B

Q2. What is the concept of delegation?

Delegation is a design pattern where an object passes on a method call to another object, effectively delegating the responsibility for handling that call. In Python, this can be achieved by calling the corresponding method on another object within the class.

Q3. What is the concept of composition?

Composition is a design principle where a class is composed of other classes as parts, and the composed classes' functionality is used to achieve the desired behaviour. It emphasizes building complex objects by combining simpler ones.

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

Bound methods are methods that are associated with an instance of a class. When a method is called on an instance, the instance is automatically passed as the first argument (usually named self). This binding of the method to an instance is what makes it a bound method.

Q5. What is the purpose of pseudoprivate attributes?

Pseudoprivate attributes in Python are attributes with double underscores (\_\_) as prefixes, such as \_\_my\_attribute. These attributes are name-mangled, meaning their names are modified to include the class name, making them less likely to clash with names in subclasses or other classes.