OOP - Inheritance & super() in Python

1. What is Inheritance?

Inheritance is an OOP principle where a class (child/subclass) acquires properties and behaviors from another class (parent/superclass). It promotes code reusability and helps create logical hierarchy.

Real-Life Analogy

A child inherits traits from their parents but can also have unique features. Similarly, in code, a child class inherits from a parent class and can extend or override it.

Basic Syntax in Python

```
class Parent:
    def greet(self):
        print("Hello from Parent")

class Child(Parent):
    def hello(self):
        print("Hello from Child")

obj = Child()
obj.greet()   # Inherited
obj.hello()   # Own method
```

2. Types of Inheritance

Multilevel Inheritance

A class inherits from another derived class, forming a chain.

Structure: A -> B -> C

```
class A:
    def method_A(self): print("A")
class B(A):
    def method_B(self): print("B")
class C(B):
    def method_C(self): print("C")
```

Multiple Inheritance

A class inherits from more than one parent class.

Structure: A + B -> C

```
class A:
    def method_A(self): print("A")
class B:
    def method_B(self): print("B")
class C(A, B):
    def method_C(self): print("C")
```

Hierarchical Inheritance

Multiple child classes inherit from a single parent class.

Structure: A -> B and A -> C

```
class A:
    def method_A(self): print("A")
class B(A):
    def method_B(self): print("B")
class C(A):
    def method_C(self): print("C")
```

OOP - Inheritance & super() in Python

3. Method Overriding

A child class can redefine a method from its parent class.

```
class Parent:
    def show(self): print("Parent")
class Child(Parent):
    def show(self): print("Child")
```

4. What is super() in Python?

super() is a built-in function (not a keyword) used to call methods from a parent class. It works based on the Method Resolution Order (MRO).

Example with super()

```
class Parent:
    def show(self): print("Parent")
class Child(Parent):
    def show(self):
        super().show()
        print("Child")
```

5. super() and MRO in Multiple Inheritance

```
class Person:
    def walk(self): print("Person Walk")

class Father(Person):
    def walk(self): print("Father Walk")
```

OOP - Inheritance & super() in Python

```
class Mother(Person):
    def walk(self): print("Mother Walk")

class Son(Mother, Father):
    pass

s = Son()
s.walk()
print(s.__mro__)
```

Output:

```
Mother Walk
(<class '__main__.Son'>, <class '__main__.Mother'>, <class '__main__.Father'>, <class '__main__.Person'>,
<class 'object'>)
```

MRO Explanation

Python follows the C3 Linearization algorithm to resolve method order.

super() follows the order defined in MRO, not just the immediate parent.