Report: Multiple Inheritance in Python

◆ 1. How super() Function Handles Multiple Inheritance

In Python, the super() function is used to call a method from the parent class in a class hierarchy. When it comes to multiple inheritance, Python uses a system called MRO (Method Resolution Order) to determine the order in which base classes are searched when calling methods via super().

The MRO ensures that each parent class is only called once, and in a consistent, predictable order. This is especially useful when different parent classes have methods with the same name.

MRO Example:

```
class A:
   def show(self):
      print("Class A")


class B(A):
   def show(self):
      print("Class B")
      super().show()


class C(A):
   def show(self):
      print("Class C")
      super().show()
```

```
class D(B, C):
    def show(self):
        print("Class D")
        super().show()


d = D()
d.show()
```

Output:

```
Class D
Class B
Class C
Class A
```

Python follows this order: D → B → C → A

🔹 2. If Human and Mammal Have the Same Method eat, How Python Handles It in Employee

If two parent classes (Human and Mammal) have a method with the same name (eat), and a child class (Employee) inherits from both, Python resolves which eat method to call based on the MRO.

<u>Example</u>:

```
class Mammal:
    def eat(self):
        print("Mammal eats")

class Human:
    def eat(self):
        print("Human eats")

class Employee(Human, Mammal):
    def eat(self):
        super().eat()
e = Employee()
e.eat()
```

Output:

```
Human eats
```

Explanation:

Since Employee inherits from Human first, Python gives priority to Human.eat() over Mammal.eat() when using super().

3- Conclusion:

super() uses the Method Resolution Order (MRO) to determine which method to call.

In multiple inheritance, Python ensures each parent is called only once using the C3 linearization algorithm.

The order of base classes in the class definition affects which method is called.

Use super() carefully to maintain consistency and avoid unexpected behavior.