Multiple Inheritance

Mahmoud Fathy Mohamed

1 What is multiple inheritance?

A class can be derived from more than one superclass in Python. This is called multiple inheritance.

For example, a class `Bat` is derived from superclasses `Mammal` and `WingedAnimal`. It makes sense because a bat is both a mammal and a winged animal.

```python
class Mammal:
    # features of Mammal class

class WingedAnimal:
    # features of WingedAnimal class

class Bat(Mammal, WingedAnimal):
    # features of Mammal + WingedAnimal + Bat classes
```

2 What will happen if the child and the parent have the same method?

Python uses Method Resolution Order (MRO) to decide which method to call, in this case the child method overrides the parents method. So, The child's method will be called.

```python
class Parent:
    def show_message(self):
        print("This is the Parent's message.")

class Child(Parent):
    def show_message(self):
        print("This is the Child's message.")

c = Child()
c.show_message() # This will call the Child's method
```

Output

```
This is the Child's message.
```

3 What will happen if two parents have the same parent?

The "Diamond Problem" is a classic issue in multiple inheritance. Python's MRO provides a clear and consistent resolution, preventing the ambiguity of which parent method to call. It specifies that methods should be inherited from the leftmost superclass when the class is called. So, here the method of the `ParentA` class is called.

```python
class Grandparent:
    def speak(self):
        print("Grandparent is speaking.")

class ParentA(Grandparent):
    def speak(self):
        print("ParentA is speaking.")

class ParentB(Grandparent):
    def speak(self):
        print("ParentB is speaking.")

class Child(ParentA, ParentB):
    pass

c = Child()
c.speak() # Python's MRO will decide to call ParentA's method.
```

Output

```
ParentA is speaking.
```