🧱 Assignment: Inheritance in Python

🎯 **Objective:**
To understand and implement different types of inheritance in Python, use the `super()` function, override methods, and apply class hierarchy design using object-oriented principles.

🟢 Level 1 – Basic Inheritance

✅ Task 1: Single Inheritance

Create a base class `Animal` with a method `speak()`. Create a derived class `Dog` that inherits from `Animal` and overrides `speak()` to print `"Bark"`.

✅ Task 2: Use of `super()`

Extend Task 1: In the `Dog` class, call the `speak()` method from the parent class using `super()` before printing `"Bark"`.

✅ Task 3: Attribute Inheritance

Create a class `Person` with attributes `name` and `age`. Create a subclass `Student` that adds a new attribute `roll_no` and a method `display_info()` to print all details.

🟡 Level 2 – Intermediate Inheritance

✅ Task 4: Multilevel Inheritance

Create the following class hierarchy:

- Class `Vehicle`
- Class `Car` inherits from `Vehicle`
- Class `ElectricCar` inherits from `Car`

Each class should have a method that prints a message unique to that class.

✅ Task 5: Hierarchical Inheritance

Create a class `Shape`. Create two subclasses `Circle` and `Square`. Each subclass should have its own method to calculate area using appropriate formulas.

✅ Task 6: Method Overriding

Create a class `Employee` with a method `work()`. Create subclasses `Manager` and `Developer` that override the `work()` method with role-specific messages.

🧱 Assignment: Inheritance in Python

🔴 Level 3 – Advanced Inheritance

✅ Task 7: Multiple Inheritance

Create two classes `Flyer` and `Swimmer`, each with a method `ability()`. Create a class `Duck` that inherits from both and overrides `ability()` to combine both messages.

✅ Task 8: Hybrid Inheritance

Design a class structure using a combination of hierarchical and multiple inheritance. Example:

- `Person` → base class
- `Student` and `Employee` → inherit from `Person`
- `Intern` → inherits from both `Student` and `Employee`

Use the `super()` function and print the full detail of an intern.

✅ Task 9: MRO Demonstration

Write a program with multiple inheritance that demonstrates the **Method Resolution Order** (MRO) using `mro()` or `__mro__` attribute.

✅ Task 10: Real-World Use Case

Create a class `BankAccount` and subclasses `SavingsAccount` and `CurrentAccount`. Each subclass should:

- Override interest calculation
- Have specific withdrawal rules
- Display account summary