**🐍 DAY 17 – HOME ASSIGNMENTS: Inheritance & Polymorphism**

(*Single, Multiple, Multilevel, Hierarchical Inheritance, Polymorphism, Method Overriding & Super Usage.*)

**🗂️ Part A: Single Inheritance**

1. Create a parent class Person with attributes:
   * name, age.
   * Method: display().
2. Create a child class Student that inherits Person:
   * Add roll\_number attribute.
   * Override the display() method.

**🌳 Part B: Multiple Inheritance**

1. Create a class Engine:
   * Method: start() returns Engine started.
2. Create a class MusicSystem:
   * Method: play() returns Playing music.
3. Create a class Car that inherits from both Engine and MusicSystem:
   * Method: drive() returns Car is driving.
   * Test by creating an object of Car and invoking methods from both parent classes.

**⛓️ Part C: Multilevel Inheritance**

1. Create a parent class Animal:
   * Method: sound() returns Generic Animal Sound.
2. Create a child class Dog inherits from Animal:
   * Override sound() returns Bark.
3. Create a grandchild class Puppy inherits from Dog:
   * Override sound() returns Yip.
4. Test by creating instances of Animal, Dog, and Puppy and invoking their sound() methods.

**🌳🌳 Part D: Hierarchical Inheritance**

1. Create a parent class Shape:
   * Attribute: name
   * Method: area() returns Not implemented
2. Create child classes:
   * Rectangle: Inherit from Shape, implement area().
   * Circle: Inherit from Shape, implement area() using math.pi.
   * Triangle: Inherit from Shape, implement area() as 0.5 \* base \* height.
3. Test instances of Rectangle, Circle, and Triangle.

**🐍 Part E: Polymorphism**

1. Create three classes:
   * Bird (method move() returns Flies),
   * Fish (method move() returns Swims),
   * Dog (method move() returns Walks).
2. Create a function test\_move(obj) that takes any of these instances and calls its move() method.
3. Create another example:
   * Add method that behaves differently:
     + Takes 2 numbers and returns the sum.
     + Takes a list of numbers and returns the total.

**⚔️ Part F: Method Overriding & Super Usage**

1. Create a parent class Employee:
   * Attribute: name, salary.
   * Method: display().
2. Create a child class Manager inherits from Employee:
   * New Attribute: department.
   * Override display() method and use super() to call parent method.
3. Create a parent class BankAccount:
   * Method: calculate\_interest() returns some value.
4. Create a child class SavingsAccount:
   * Override the method to return a different interest rate.

**⚡️ Part G: Complex Challenges**

1. **Shape Manager**:
   * Create a parent class Shape with a method draw().
   * Create child classes Square, Rectangle, Circle, and implement their draw() method.
   * Maintain a list of shapes and iterate over them, calling draw().
2. **Employee Hierarchy Simulation**:
   * Create parent class Person (name, age).
   * Create child classes:
     + Employee inherits Person (add salary).
     + Teacher inherits Employee (add subject).
     + Engineer inherits Employee (add specialization).
   * Implement a display() method in each class.
3. **Geometric Calculator**:
   * Create parent class Polygon (attribute sides).
   * Create child classes:
     + Square: area = side^2.
     + Rectangle: area = length \* breadth.
     + Triangle: area = (base \* height)/2.
   * Polymorphically call area methods.
4. **Zoo Simulation**:
   * Create parent class Animal:
     + Methods: sound(), move().
   * Create child classes:
     + Lion: sound() -> Roar.
     + Snake: move() -> Slither.
     + Parrot: sound() -> Talk.
   * Maintain a list of animals and call their methods.
5. **Custom Exceptions with Inheritance**:
   * Create a parent exception BankError.
   * Create child exceptions:
     + InsufficientBalanceError.
     + InvalidAmountError.
   * Simulate a bank withdrawal method that raises these errors appropriately.
6. **Library System**:
   * Create parent class LibraryItem.
   * Create child classes:
     + Book: title, author.
     + Magazine: title, issue\_number.
     + DVD: title, duration.
   * Implement a method display() for each.