AI-2002 Lab 2 Tasks

Task 1: Student Profile System (Basics of Class & Object)

Create a Student class with attributes name, roll_no, and program.
Add a method display_info() to show student details.
Create at least **two student objects** and display their information.

Task 2: Smart Fan Controller (Constructor & Methods)

Create a SmartFan class with:

- Attributes: room_name, speed
- Methods: increase_speed(), decrease_speed(), display_status()

Create multiple fan objects and change their speed.

Task 3: Library Book System (Class vs Instance Attributes)

Create a Book class with:

- Class attribute: library_name
- Instance attributes: title, author

Add a method display_book() and show how changing the **class attribute** affects all objects.

Task 4: Bank Account Manager (Methods & Attributes)

Create a BankAccount class with:

- Attributes: account_holder, account_no, balance
- Methods: deposit(amount), withdraw(amount), display_account()

Create multiple accounts and perform transactions.

AI-2002 Lab 2 Tasks

Task 5: Smart Home Light System (OOP Automation)
Create a SmartLight class with:

- Attributes: room, status
- Methods: turn_on(), turn_off(), show_status()

Create lights for different rooms and control them.

Task 6: University Staff System (Inheritance)
Create a base class Staff with:

- Attributes: name, staff_id, department
- Method: display_info()

Create subclasses:

- Teacher (courses, salary)
- AdminStaff (role, working_hours)
- ResearchAssistant (research_topic, stipend)

Override display_info() where needed.

Task 7: Vehicle System (Method Overriding – Polymorphism)

Create a base class Vehicle with method start().
Create subclasses Car, Bike, and Bus that override start().

Create objects and call start() to observe polymorphism.

Task 8: Calculator Utility (Method Overloading Simulation)

Create a Calculator class with method multiply() using:

- Default arguments
- *args

Allow multiplication of 2, 3, or more numbers.

AI-2002 Lab 2 Tasks

Task 9: Employee Payroll System (Protected Members)

Task:
Create an Employee class with protected attributes _name and _salary.
Create a subclass Manager that accesses and displays these protected members.

Task 10: Secure Banking System (Encapsulation)

Create a BankAccount class with:

- Private attribute __balance
- Public methods: deposit(), withdraw(), get_balance()

Demonstrate that __balance **cannot be accessed directly** from outside the class.