Assignment: Object-Oriented Programming

Title: *Designing an Animal Shelter Management System*

Objective:
To implement an object-oriented program that uses multiple object types to demonstrate OOP principles such as encapsulation, inheritance, polymorphism, and abstraction.

Problem Statement

Develop a program to manage an **Animal Shelter**. The system should handle different types of animals (e.g., **Dogs**, **Cats**, **Birds**) and their interactions with staff members and adopters. The program should track animal information, staff activities, and adoption processes.

Requirements

1. Classes to Implement:

- **Animal** (Base Class)
    - Attributes: `name`, `species`, `age`, `health_status`, `adoption_status`
    - Methods:
        - `display_info()`: Display the animal's details.
        - `update_health_status(status)`: Update the health of the animal.
- **Dog** and **Cat** (Derived Classes of Animal)
    - Additional Attributes:
        - For `Dog`: `breed`, `trained` (boolean)
        - For `Cat`: `color`, `indoor` (boolean)
    - Override `display_info()` to include specific details.
- **Bird** (Derived Class of Animal)
    - Additional Attributes: `wing_span`, `can_fly` (boolean)
    - Override `display_info()` to include specific details.
- **Staff**
    - Attributes: `staff_id`, `name`, `role`, `tasks`
    - Methods:
        - `assign_task(task)`: Assign a task to a staff member.

        - `display_tasks()`: Display assigned tasks.
- **Adopter**
    - Attributes: `adopter_id`, `name`, `contact_info`, `adopted_animals` (list)
    - Methods:
        - `adopt_animal(animal)`: Add an animal to the list of adopted animals.
        - `display_adopted_animals()`: Display a list of animals adopted by the adopter.

2. Functional Requirements:

- **Adding Animals:**
    - Allow staff to add animals to the system (as `Dog`, `Cat`, or `Bird`).
- **Viewing Animals:**
    - Display all animals, grouped by type.
    - Include filters for available or adopted animals.
- **Adoption Process:**
    - Enable an adopter to adopt an animal, updating its status.
- **Health Updates:**
    - Allow staff to update the health status of animals.

3. Program Features:

- Use **Inheritance** to define relationships between the base `Animal` class and its subclasses.
- Apply **Polymorphism** in methods like `display_info()` to handle different animal types.
- Encapsulate attributes using private or protected access and provide getter/setter methods where necessary.
- Demonstrate interactions between `Animal`, `Staff`, and `Adopter` objects.

Submission

- Code Implementation:
    - Upload your source code file(s).
    - Ensure the code is properly commented.
- Write-up:
    - Describe the OOP principles used in your solution.

    - Provide examples of how these principles are implemented in your code.

Sample Output

1. Add a new Dog with details: `name=Buddy`, `species=Dog`, `age=3`, `breed=Labrador`, `trained=True`.
2. Display all animals: Output shows "Buddy - Labrador - Available."
3. Adopter adopts Buddy. Status updates to "Adopted."
4. Staff updates Buddy’s health status to "Healthy."
