Write a Java program to create a class called "Pet" with attributes for name, species, and age. Create subclasses "Dog" and "Bird" that add specific attributes like favorite toy for dogs and wing span for birds. Implement methods to display pet details and calculate the pet's age in human years.

Sample Solution:

Java Code:

Pet.java

```
// Define the Pet class
class Pet {
    // Attributes for the Pet class
    String name;
    String species;
    int age;

    // Constructor for the Pet class
    public Pet(String name, String species, int age) {
        this.name = name;
        this.species = species;
        this.age = age;
    }

    // Method to display pet details
    public void displayDetails() {
        System.out.println("Name: " + name);
        System.out.println("Species: " + species);
        System.out.println("Age: " + age + " years");
    }

    // Method to calculate pet's age in human years
    public int calculateHumanAge() {
        if (species.equals("Dog")) {
            return age * 7;
        } else {
            // Assuming bird's age in human years is the same as actual age
            return age;
        }
    }
```

```
    }
```

Explanation:

- Class definition: Defines the Pet class.
- Attributes: Declares three attributes: name (String), species (String), and age (int).
- Constructor: Initializes the name, species, and age attributes with provided values.
- displayDetails() method: Prints the pet's name, species, and age to the console.
- calculateHumanAge() method:
    - Check if the species is "Dog".
    - If true, returns the pet's age multiplied by 7 (dog's age in human years).
    - Otherwise, returns the actual age (assuming birds' age in human years is the same as their actual age).

Dog.java

```
// Define the Dog class that extends Pet
class Dog extends Pet {
    // Additional attribute for Dog
    String favoriteToy;

    // Constructor for the Dog class
    public Dog(String name, int age, String favoriteToy) {
        super(name, "Dog", age); // Call the constructor of the superclass
        this.favoriteToy = favoriteToy;
    }

    // Override the displayDetails method to include favorite toy
    @Override
    public void displayDetails() {
        super.displayDetails();
        System.out.println("Favorite Toy: " + favoriteToy);
    }
}
```

Explanation:

- Class definition: Defines the Dog class, which extends the Pet class.

- Additional attribute: Declares favoriteToy (String) specific to the Dog class.
- Constructor:
    - Initializes the name, age, and favoriteToy attributes.
    - Override displayDetails() method:
        - Calls the displayDetails() method of the Pet class.
        - Adds a line to print the dog's favorite toy.

Bird.java

```
// Define the Bird class that extends Pet
class Bird extends Pet {
    // Additional attribute for Bird
    double wingSpan;

    // Constructor for the Bird class
    public Bird(String name, int age, double wingSpan) {
        super(name, "Bird", age); // Call the constructor of the superclass
        this.wingSpan = wingSpan;
    }

    // Override the displayDetails method to include wing span
    @Override
    public void displayDetails() {
        super.displayDetails();
        System.out.println("Wing Span: " + wingSpan + " meters");
    }
}
```

Explanation:

- Class definition: Defines the Bird class, which extends the Pet class.
- Additional attribute: Declares wingSpan (double) specific to the Bird class.
- Constructor:
    - Initializes the name, age, and wingSpan attributes.
    - Calls the superclass constructor with name, "Bird" (as species), and age.
- Override displayDetails() method:
    - Calls the displayDetails() method of the Pet class.

- Add a line to print the bird's wing span in meters.

Main.java

```
// Main class to test the Pet, Dog, and Bird classes
public class Main {
    public static void main(String[] args) {
        // Create an instance of Dog
        Dog dog = new Dog("Cooper", 3, "Ball");

        // Create an instance of Bird
        Bird bird = new Bird("Pelican", 2, 0.5);

        // Display details of the dog
        System.out.println("Dog Details:");
        dog.displayDetails();
        System.out.println("Dog's age in human years: " + dog.calculateHuma

        // Display details of the bird
        System.out.println("\nBird Details:");
        bird.displayDetails();
        System.out.println("Bird's age in human years: " + bird.calculateHu
    }
}
```

Explanation:

- Class definition: Defines the Main class to test the Pet, Dog, and Bird classes.
- main method: Contains the main logic for testing.
  - Create an instance of Dog: Instantiates a Dog object named "Cooper" with age 3 and favorite toy "Ball".
  - Create an instance of Bird: Instantiates a Bird object named "Pelican" with age 2 and a wing span of 0.5 meters.
  - Display details of the dog:
    - Prints "Dog Details:" to the console.
    - Calls dog.displayDetails() to print the dog's details.
    - Calls dog.calculateHumanAge() to print the dog's age in human years.
  - Display details of the bird:

- Prints "\nBird Details:" to the console.
- Calls bird.displayDetails() to print the bird's details.
- Calls bird.calculateHumanAge() to print the bird's age in human years.

Output:

```
Dog Details:
Name: Cooper
Species: Dog
Age: 3 years
Favorite Toy: Ball
Dog's age in human years: 21

Bird Details:
Name: Pelican
Species: Bird
Age: 2 years
Wing Span: 0.5 meters
Bird's age in human years: 2
```

Java Code Editor: