```java
// Base interface Animal
interface Animal {
    void eat();
}

// Extended interface Pet
interface Pet extends Animal {
    void play();
}

// Dog class implementing Pet (and indirectly Animal)
class Dog implements Pet {
    private String name;

    public Dog(String name) {
        this.name = name;
    }

    @Override
    public void eat() {
        System.out.println(name + " is eating dog food.");
    }

    @Override
    public void play() {
        System.out.println(name + " is playing fetch.");
    }

    public void showDetails() {
        System.out.println("Dog Name: " + name);
        eat();
        play();
    }
}

// Main class to test the implementation
public class InterfaceInheritanceDemo {
    public static void main(String[] args) {
        Dog myDog = new Dog("Bruno");
        myDog.showDetails();
```

```
    }
}
```

```
PS E:\JAVA PROGRAMS\steparyansingh\year2\oops\week8\assignment>
>> java InterfaceInheritanceDemo
Dog Name: Bruno
Bruno is eating dog food.
Bruno is playing fetch.
PS E:\JAVA PROGRAMS\steparyansingh\year2\oops\week8\assignment>
```

```java
// Abstract class Shape
abstract class Shape {
    // Abstract methods
    public abstract double area();
    public abstract double perimeter();

    // Concrete method
    public void displayInfo() {
        System.out.println("This is a geometric shape.");
    }
}

// Circle subclass
class Circle extends Shape {
    private double radius;

    public Circle(double radius) {
        this.radius = radius;
    }
```

```
    @Override
    public double area() {
        return Math.PI * radius * radius;
    }

    @Override
    public double perimeter() {
        return 2 * Math.PI * radius;
    }

    public void showDetails() {
        displayInfo();
        System.out.println("Shape: Circle");
        System.out.println("Radius: " + radius);
        System.out.println("Area: " + area());
        System.out.println("Perimeter: " + perimeter());
    }
}

// Rectangle subclass
class Rectangle extends Shape {
    private double length;
    private double width;

    public Rectangle(double length, double width) {
        this.length = length;
        this.width = width;
    }

    @Override
    public double area() {
        return length * width;
    }

    @Override
    public double perimeter() {
        return 2 * (length + width);
    }

    public void showDetails() {
```

```
        displayInfo();
        System.out.println("Shape: Rectangle");
        System.out.println("Length: " + length + ", Width: " + width);
        System.out.println("Area: " + area());
        System.out.println("Perimeter: " + perimeter());
    }
}

// Main class to test the implementation
public class ShapeInfoDemo {
    public static void main(String[] args) {
        Circle circle = new Circle(4.5);
        Rectangle rectangle = new Rectangle(5.0, 3.0);

        circle.showDetails();
        System.out.println();
        rectangle.showDetails();
    }
}
```

```
PS E:\JAVA_PROGRAMS\steparyansingh\year2\oops\week8\assignment> java ShapeInfoDemo
Shape: Circle
Radius: 4.5
Area: 63.61725123519331
Perimeter: 28.274333882308138

This is a geometric shape.
Shape: Rectangle
Length: 5.0, Width: 3.0
Area: 15.0
Perimeter: 16.0
PS E:\JAVA_PROGRAMS\steparyansingh\year2\oops\week8\assignment>
```

```java
// Interface Playable
interface Playable {
    void play();
    void pause();
}

// MusicPlayer class implementing Playable
class MusicPlayer implements Playable {
    @Override
    public void play() {
        System.out.println("Playing music...");
    }

    @Override
    public void pause() {
        System.out.println("Music paused.");
    }
}

// VideoPlayer class implementing Playable
class VideoPlayer implements Playable {
    @Override
    public void play() {
        System.out.println("Playing video...");
    }

    @Override
    public void pause() {
        System.out.println("Video paused.");
    }
}

// Main class to demonstrate polymorphism
public class PlayableDemo {
    public static void main(String[] args) {
        Playable ref;

        ref = new MusicPlayer();
        ref.play();
        ref.pause();
```

```
        System.out.println();

        ref = new VideoPlayer();
        ref.play();
        ref.pause();
    }
}
```

```
PS E:\JAVA PROGRAMS\steparyansingh\year2\oops\week8\assignment> java PlayableDemo
Playing music...
Music paused.

Playing video...
Video paused.
PS E:\JAVA PROGRAMS\steparyansingh\year2\oops\week8\assignment>
```

```
// Abstract class Vehicle
abstract class Vehicle {
    // Abstract method
    public abstract void start();

    // Concrete method
    public void stop() {
        System.out.println("Vehicle stopped.");
    }
}

// Interface Fuel
interface Fuel {
    void refuel();
}

// Car class extending Vehicle and implementing Fuel
class Car extends Vehicle implements Fuel {
```

```
    private String model;

    public Car(String model) {
        this.model = model;
    }

    @Override
    public void start() {
        System.out.println(model + " is starting...");
    }

    @Override
    public void refuel() {
        System.out.println(model + " is refueling with petrol.");
    }

    public void showDetails() {
        System.out.println("Car Model: " + model);
        start();
        refuel();
        stop();
    }
}

// Main class to test the implementation
public class VehicleFuelDemo {
    public static void main(String[] args) {
        Car myCar = new Car("Swift");
        myCar.showDetails();
    }
}
```

```
Video paused.
PS E:\JAVA PROGRAMS\steparyansingh\year2\oops\week8\assignment> java VehicleFuelDemo
Car Model: Swift
Swift is starting...
Swift is refueling with petrol.
Vehicle stopped.
PS E:\JAVA PROGRAMS\steparyansingh\year2\oops\week8\assignment>
```