```java
import java.util.Scanner;

interface Animal {
    void eat();

    void sleep();

    void move();

    void reproduce();

    void communicate();
}

abstract class Bird implements Animal {
    private String species;

    public Bird(String species) {
        this.species = species;
        // System.out.println(species + " is a bird");
    }

    public void move() {
        System.out.println(species + " moves like a bird");
    }

    abstract void fly();

    public String getSpecies() {
        return species;
    }
}

abstract class Reptile implements Animal {
    private String species;

    public Reptile(String species) {
        this.species = species;
        System.out.println(species + " is a reptile");
    }

    public void move() {
        System.out.println(species + " moves like a reptile");
    }

    abstract void swim();

    public String getSpecies() {
        return species;
    }
}

abstract class Mammal implements Animal {
    private String species;
```

```
    public Mammal(String species) {
        this.species = species;
        System.out.println(species + " is a mammal");
    }

    public void move() {
        System.out.println(species + " moves like a mammal");
    }

    abstract void walk();

    public String getSpecies() {
        return species;
    }
}

class Penguin extends Bird {
    public Penguin() {
        super("Penguin");
    }

    public void eat() {
        System.out.println(getSpecies() + " eats fish");
    }

    public void sleep() {
        System.out.println(getSpecies() + " sleeps standing");
    }

    public void reproduce() {
        System.out.println(getSpecies() + " lays eggs");
    }

    public void communicate() {
        System.out.println(getSpecies() + " makes sounds");
    }

    public void fly() {
        System.out.println(getSpecies() + " cannot fly");
    }

    public void swim() {
        System.out.println(getSpecies() + " swims well");
    }
}

class Sparrow extends Bird {
    public Sparrow() {
        super("Sparrow");
    }

    public void eat() {
        System.out.println(getSpecies() + " eats seeds");
```

```
    }

    public void sleep() {
        System.out.println(getSpecies() + " sleeps in nest");
    }

    public void reproduce() {
        System.out.println(getSpecies() + " lays eggs");
    }

    public void communicate() {
        System.out.println(getSpecies() + " chirps");
    }

    public void fly() {
        System.out.println(getSpecies() + " flies fast");
    }
}

class Crocodile extends Reptile {
    public Crocodile() {
        super("Crocodile");
    }

    public void eat() {
        System.out.println(getSpecies() + " eats meat");
    }

    public void sleep() {
        System.out.println(getSpecies() + " sleeps with eyes open");
    }

    public void reproduce() {
        System.out.println(getSpecies() + " lays eggs");
    }

    public void communicate() {
        System.out.println(getSpecies() + " growls");
    }

    public void swim() {
        System.out.println(getSpecies() + " swims stealthily");
    }
}

class Human extends Mammal {
    public Human() {
        super("Human");
    }

    public void eat() {
        System.out.println(getSpecies() + " eats varied diet");
    }
```

```
    public void sleep() {
        System.out.println(getSpecies() + " sleeps lying down");
    }

    public void reproduce() {
        System.out.println(getSpecies() + " gives birth");
    }

    public void communicate() {
        System.out.println(getSpecies() + " speaks languages");
    }

    public void walk() {
        System.out.println(getSpecies() + " walks upright");
    }
}

public class AnimalDemoNew {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Display knowledge table
        String[] table = {
                "---------------------------------------",
                "|  Subject   |  Object  |  Predicate  |",
                "|------------|----------|-------------|",
                "|            |  bird    |  True       |",
                "|  Penguin   |  fly     |  False      |",
                "|            |  eat     |  Fish       |",
                "|------------|----------|-------------|",
                "|            |  reptile |  True       |",
                "|  Crocodile |  swim    |  True       |",
                "|            |  eat     |  Meat       |",
                "|------------|----------|-------------|",
                "|            |  mammal  |  True       |",
                "|  Human     |  walk    |  True       |",
                "|            |  eat     |  Varied     |",
                "|------------|----------|-------------|",
                "|            |  bird    |  True       |",
                "|  Sparrow   |  fly     |  True       |",
                "|            |  eat     |  Seeds      |",
                "---------------------------------------"
        };
        for (String line : table)
            System.out.println(line);

        while (true) {
            System.out.print("\nAsk a question (e.g., 'Penguin is bird?') or type 'quit':
");
            String question = sc.nextLine().trim();

            String[] parts = question.split(" ");
            if (parts.length < 3) {
                return;
```

```
            }

            String subject = parts[0].toLowerCase();
            String verb = parts[1].toLowerCase();
            String object = parts[2].toLowerCase().replace("?", "");

            Animal animal = null;
            switch (subject) {
                case "penguin":
                    animal = new Penguin();
                    if ((verb.equals("is") && object.equals("bird")) && animal instanceof
Bird) {
                        System.out.println("Yes, Penguin is a bird.");
                    } else if (verb.equals("can") && object.equals("fly")) {
                        ((Penguin) animal).fly();
                    } else if (verb.equals("can") && object.equals("swim")) {
                        ((Penguin) animal).swim();
                    } else if (verb.equals("eat")) {
                        ((Penguin) animal).eat();
                    } else {
                        System.out.println("I don't know the answer to that.");
                    }
                    break;

                case "sparrow":
                    animal = new Sparrow();
                    if ((verb.equals("is") && object.equals("bird")) && animal instanceof
Bird) {
                        System.out.println("Yes, Sparrow is a bird.");
                    } else if (verb.equals("can") && object.equals("fly")) {
                        ((Sparrow) animal).fly();
                    } else if (verb.equals("eat")) {
                        animal.eat();
                    }
                    break;

                case "crocodile":
                    animal = new Crocodile();
                    if ((verb.equals("is") && object.equals("reptile")) && animal
instanceof Reptile) {
                        System.out.println("Yes, Crocodile is a reptile.");
                    } else if (verb.equals("can") && object.equals("swim")) {
                        ((Crocodile) animal).swim();
                    } else if (verb.equals("eat")) {
                        animal.eat();
                    }
                    break;

                case "human":
                    animal = new Human();
                    if ((verb.equals("is") && object.equals("mammal")) && animal
instanceof Mammal) {
                        System.out.println("Yes, Human is a mammal.");
                    } else if (verb.equals("can") && object.equals("walk")) {
```

```
                        ((Human) animal).walk();
                    } else if (verb.equals("eat")) {
                        animal.eat();
                    }
                    break;

                case "quit":
                    System.out.println("Exiting the program.");
                    return;

                default:
                    System.out.println("Unknown animal.");
            }
        }
    }
}
```