**Exercise 1:**

**Create an interface named Shape with a method calculateArea(). Implement this interface in two classes: Circle and Rectangle. Calculate the area of each shape.**

**Answer 1:**

interface Shape {

double calculateArea();

}

class Circle implements Shape {

private double radius;

public Circle(double radius) {

this.radius = radius;

}

@Override

public double calculateArea() {

return Math.PI \* radius \* radius;

}

}

class Rectangle implements Shape {

private double length;

private double width;

public Rectangle(double length, double width) {

this.length = length;

this.width = width;

}

@Override

public double calculateArea() {

return length \* width;

}

}

**Exercise 2:**

**Create an interface named Vehicle with methods start() and stop(). Implement this interface in two classes: Car and Motorcycle.**

**Answer 2:**

interface Vehicle {

void start();

void stop();

}

class Car implements Vehicle {

@Override

public void start() {

System.out.println("Car started.");

}

@Override

public void stop() {

System.out.println("Car stopped.");

}

}

class Motorcycle implements Vehicle {

@Override

public void start() {

System.out.println("Motorcycle started.");

}

@Override

public void stop() {

System.out.println("Motorcycle stopped.");

}

}

**Exercise 3:**

**Create an interface named Calculator with methods add() and subtract(). Implement this interface in a class named BasicCalculator.**

**Answer 3:**

interface Calculator {

int add(int a, int b);

int subtract(int a, int b);

}

class BasicCalculator implements Calculator {

@Override

public int add(int a, int b) {

return a + b;

}

@Override

public int subtract(int a, int b) {

return a - b;

}

}

**Exercise 4:**

**Create an interface named Playable with a method playSound(). Implement this interface in classes Piano and Guitar.**

**Answer 4:**

interface Playable {

void playSound();

}

class Piano implements Playable {

@Override

public void playSound() {

System.out.println("Piano sound is played.");

}

}

class Guitar implements Playable {

@Override

public void playSound() {

System.out.println("Guitar sound is played.");

}

}

**Exercise 5:**

**Create an interface named Logger with a method logMessage(String message). Implement this interface in classes ConsoleLogger (logs to console) and FileLogger (logs to a file).**

**Answer 5:**

interface Logger {

void logMessage(String message);

}

class ConsoleLogger implements Logger {

@Override

public void logMessage(String message) {

System.out.println("Console Logger: " + message);

}

}

class FileLogger implements Logger {

@Override

public void logMessage(String message) {

// Code to log message to a file

System.out.println("File Logger: " + message);

}

}

**Exercise 6:**

**Create an interface named Drawable with a method draw(). Implement this interface in classes Circle and Square.**

**Answer 6:**

interface Drawable {

void draw();

}

class Circle implements Drawable {

@Override

public void draw() {

System.out.println("Drawing a circle.");

}

}

class Square implements Drawable {

@Override

public void draw() {

System.out.println("Drawing a square.");

}

}

**Exercise 7:**

**Create an interface named Animal with a method makeSound(). Implement this interface in classes Dog and Cat.**

**Answer 7:**

interface Animal {

void makeSound();

}

class Dog implements Animal {

@Override

public void makeSound() {

System.out.println("Dog barks.");

}

}

class Cat implements Animal {

@Override

public void makeSound() {

System.out.println("Cat meows.");

}

}

**Exercise 8:**

**Create an interface named Database with methods connect() and disconnect(). Implement this interface in classes MySQLDatabase and OracleDatabase.**

**Answer 8:**

interface Database {

void connect();

void disconnect();

}

class MySQLDatabase implements Database {

@Override

public void connect() {

System.out.println("Connected to MySQL database.");

}

@Override

public void disconnect() {

System.out.println("Disconnected from MySQL database.");

}

}

class OracleDatabase implements Database {

@Override

public void connect() {

System.out.println("Connected to Oracle database.");

}

@Override

public void disconnect() {

System.out.println("Disconnected from Oracle database.");

}

}

**Exercise 9:**

**Create an interface named Employee with methods calculateSalary() and displayInfo(). Implement this interface in classes Manager and Developer.**

**Answer 9:**

interface Employee {

double calculateSalary();

void displayInfo();

}

class Manager implements Employee {

@Override

public double calculateSalary() {

// Calculation logic for manager's salary

return 50000.0;

}

@Override

public void displayInfo() {

System.out.println("Manager's information.");

}

}

class Developer implements Employee {

@Override

public double calculateSalary() {

// Calculation logic for developer's salary

return 60000.0;

}

@Override

public void displayInfo() {

System.out.println("Developer's information.");

}

}

**Exercise 10:**

**Create an interface named Resizable with a method resize(double factor). Implement this interface in a class named ResizableCircle.**

**Answer 10:**

interface Resizable {

void resize(double factor);

}

class ResizableCircle implements Resizable {

private double radius;

public ResizableCircle(double radius) {

this.radius = radius;

}

@Override

public void resize(double factor) {

radius \*= factor;

}

// Other methods and properties for the circle

}