Practical 08

Exercise 01:

Create a class named "BankAccount" with private instance variables "accountNumber" and "balance." Implement encapsulation by providing public getter and setter methods for both variables. Additionally, create an abstract method called "calculateInterest" in the "BankAccount" class. Implement two subclasses, "SavingsAccount" and "CheckingAccount," that extend the "BankAccount" class and provide their own implementations of the "calculateInterest" method. Write the implementation code for the getter and setter methods in the "BankAccount" class, and the "calculateInterest" method in both the "SavingsAccount" and "CheckingAccount" classes. Assuming that the interest for saving is 12% and checking is 2% (both private variables), find out What will be the interest for a person with 1 million in his checking and 20 million in his saving account.

```
abstract class BankAccount {
  private String accountNumber;
  private double balance;

  public String getAccountNumber() {
    return accountNumber;
  }

  public void setAccountNumber(String accountNumber) {
    this.accountNumber = accountNumber;
  }

  public double getBalance() {
    return balance;
  }

  public void setBalance(double balance) {
    this.balance = balance;
  }

  // Abstract method to calculate interest, to be implemented by subclasses
  public abstract double calculateInterest();
}

class SavingsAccount extends BankAccount {
  private final double savingsInterestRate = 0.12; // 12% interest rate for savings account

  @Override
  public double calculateInterest() {
    return getBalance() * savingsInterestRate;
  }
}

class CheckingAccount extends BankAccount {
  private final double checkingInterestRate = 0.02; // 2% interest rate for checking account
```

```
    @Override
    public double calculateInterest() {
        return getBalance() * checkingInterestRate;
    }
}

public class Main {
    public static void main(String[] args) {
        SavingsAccount savingsAccount = new SavingsAccount();
        savingsAccount.setBalance(20000000); // 20 million
        double savingsInterest = savingsAccount.calculateInterest();

        CheckingAccount checkingAccount = new CheckingAccount();
        checkingAccount.setBalance(1000000); // 1 million
        double checkingInterest = checkingAccount.calculateInterest();

        System.out.println("Interest for Savings Account: $" + savingsInterest);
        System.out.println("Interest for Checking Account: $" + checkingInterest);
    }
}
```

Exercise 02:

Create an interface called "Shape" with two abstract methods: "double calculateArea()" and "double calculatePerimeter()". Implement the "Shape" interface in three classes: "Circle", "Rectangle", and "Triangle". Each class should have private instance variables relevant to its shape, and provide public getter and setter methods for these variables. Additionally, each class should define a constructor that initializes the instance variables. Write the implementation code for the "Shape" interface, the getter and setter methods in each class, and the constructors in each class.

```
// Shape interface
interface Shape {
    double calculateArea();
    double calculatePerimeter();
}

// Circle class
class Circle implements Shape {
    private double radius;

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

    public void setRadius(double radius) {
        this.radius = radius;
    }

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

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

// Rectangle class
class Rectangle implements Shape {
    private double length;
    private double width;

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

    public double getLength() {
        return length;
    }

    public void setLength(double length) {
        this.length = length;
    }

    public double getWidth() {
        return width;
    }
    public void setWidth(double width) {
        this.width = width;
    }

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

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

// Triangle class
class Triangle implements Shape {
    private double side1;
    private double side2;
    private double side3;

    public Triangle(double side1, double side2, double side3) {
        this.side1 = side1;
        this.side2 = side2;
        this.side3 = side3;
    }

    public double getSide1() {
        return side1;
    }

    public void setSide1(double side1) {
        this.side1 = side1;
    }

    public double getSide2() {
        return side2;
    }

    public void setSide2(double side2) {
        this.side2 = side2;
    }

    public double getSide3() {
        return side3;
    }

    public void setSide3(double side3) {
        this.side3 = side3;
    }

    @Override
    public double calculateArea() {
        // Using Heron's formula to calculate area of a triangle
        double s = (side1 + side2 + side3) / 2;
        return Math.sqrt(s * (s - side1) * (s - side2) * (s - side3));
    }

    @Override
    public double calculatePerimeter() {
        return side1 + side2 + side3;
    }
}

public class Main {
    public static void main(String[] args) {
        Circle circle = new Circle(5);
        System.out.println("Circle Area: " + circle.calculateArea());
        System.out.println("Circle Perimeter: " + circle.calculatePerimeter());

        Rectangle rectangle = new Rectangle(4, 6);
        System.out.println("Rectangle Area: " + rectangle.calculateArea());
        System.out.println("Rectangle Perimeter: " + rectangle.calculatePerimeter());

        Triangle triangle = new Triangle(3, 4, 5);
        System.out.println("Triangle Area: " + triangle.calculateArea());
        System.out.println("Triangle Perimeter: " + triangle.calculatePerimeter());
    }
}
```