BankAccount.java:

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

    // Getter and Setter for accountNumber
    public String getAccountNumber() {
        return accountNumber;
    }

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

    // Getter and Setter for balance
    public double getBalance() {
        return balance;
    }

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

    // Abstract method calculateInterest to be implemented in subclasses
    public abstract double calculateInterest();
}
```

SavingsAccount.java:

```
public class SavingsAccount extends BankAccount {
    private final double INTEREST_RATE = 0.12; // 12% interest

    @Override
```

```java
    public double calculateInterest() {
        return getBalance() * INTEREST_RATE;
    }
}
```

CheckingAccount.java:

```java
public class CheckingAccount extends BankAccount {
    private final double INTEREST_RATE = 0.02; // 2% interest

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

TestBankAccount.java:

```java
public class TestBankAccount {
    public static void main(String[] args) {
        CheckingAccount checkingAccount = new CheckingAccount();
        SavingsAccount savingsAccount = new SavingsAccount();

        checkingAccount.setAccountNumber("123456");
        checkingAccount.setBalance(1000000);

        savingsAccount.setAccountNumber("789012");
        savingsAccount.setBalance(20000000);

        double checkingInterest = checkingAccount.calculateInterest();
        double savingsInterest = savingsAccount.calculateInterest();

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

Output:

```
Interest for Checking Account: $20000.0
Interest for Savings Account: $2400000.0
```

Shape.java (Interface):

```java
public interface Shape {
    double calculateArea();
    double calculatePerimeter();
}
```

Circle.java:

```java
public class Circle implements Shape {
    private double radius;

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

    // Getter and Setter for radius
    public double getRadius() {
        return radius;
    }

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

    // Implementing Shape interface methods
    @Override
    public double calculateArea() {
        return Math.PI * radius * radius;
    }

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

Rectangle.java:

```
public class Rectangle implements Shape {
    private double length;
    private double width;

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

    // Getters and Setters for length and 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;
    }

    // Implementing Shape interface methods
    @Override
    public double calculateArea() {
        return length * width;
    }

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

Triangle.java:

```
public class Triangle implements Shape {
    private double side1;
    private double side2;
    private double side3;

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

    // Getters and Setters for sides
    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;
}

// Implementing Shape interface methods
@Override
public double calculateArea() {
    // Assuming the implementation for area of triangle is already known
    // This can be implemented using Heron's formula or other methods.
    // For simplicity, we'll return 0 here.
    return 0;
}
```

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

public class TestShapes {
    public static void main(String[] args) {
        Circle circle = new Circle(5);
        Rectangle rectangle = new Rectangle(4, 6);
        Triangle triangle = new Triangle(3, 4, 5);

        // Get and print area and perimeter for each shape
        System.out.println("Circle Area: " + circle.calculateArea());
        System.out.println("Circle Perimeter: " + circle.calculatePerimeter());

        System.out.println("Rectangle Area: " + rectangle.calculateArea());
        System.out.println("Rectangle Perimeter: " + rectangle.calculatePerimeter());

        System.out.println("Triangle Area: " + triangle.calculateArea());
        System.out.println("Triangle Perimeter: " + triangle.calculatePerimeter());
    }
}
```