Exercício 1: No material da “Aula 07 – Relacionamento entre Classes”, no tópico “Relacionamento entre classes do tipo É-UM”, na página 7, implemente a classe para testes das classes listadas nessa página.

```
public class Main {

      public static class Animal {
      public void mover() {
          System.out.println("O animal está se movendo");
       }

          public void emitirSom() {
          System.out.println("O animal emite um som genérico");
       }
   }


   public static class Cachorro extends Animal {

       public void latir() {
          System.out.println("O cachorro está latindo: Au au");
       }


       @Override
       public void emitirSom() {
          System.out.println("O cachorro emite som: Au au");
       }
   }


   public static class Gato extends Animal {

       public void miar() {
          System.out.println("O gato está miando: Miau");
       }


       @Override
       public void emitirSom() {
          System.out.println("O gato emite som: Miau");
       }
   }


   public static void main(String[] args) {

       Cachorro cachorro = new Cachorro();
       System.out.println("Testando o Cachorro:");
       cachorro.mover();
       cachorro.latir();
```

```
        cachorro.emitirSom();

        System.out.println();


        Gato gato = new Gato();
        System.out.println("Testando o Gato:");
        gato.mover();
        gato.miar();
        gato.emitirSom();
    }
}
```

Exercício 2: No material da “Aula 07 – Relacionamento entre Classes”, no tópico “Classes Abstratas”, na página 13, implemente a classe para testes das classes listadas nessa página.

```
public class Main {

        public static abstract class Animal {

        public void mover() {
        System.out.println("O animal está se movendo");
        }


        public abstract void emitirSom();
    }


    public static class Cachorro extends Animal {

        @Override
        public void emitirSom() {
            System.out.println("O cachorro emite som: Au au");
        }
    }


    public static class Gato extends Animal {

        @Override
        public void emitirSom() {
            System.out.println("O gato emite som: Miau");
        }
    }


    public static void main(String[] args) {
```

```
        Animal cachorro = new Cachorro();
        System.out.println("Testando o Cachorro:");
        cachorro.mover();
        cachorro.emitirSom();

        System.out.println();


        Animal gato = new Gato();
        System.out.println("Testando o Gato:");
        gato.mover();
        gato.emitirSom();
    }
}
```

Exercício 3: No material da “Aula 07 – Relacionamento entre Classes”, no tópico “Interfaces”, na página 18, implemente a classe para testes das classes listadas nessa página.

```
public interface FormaGeometrica {
    double calcularArea();
    double calcularPerimetro();
}


public class Retangulo implements FormaGeometrica {
    private double largura;
    private double altura;

    public Retangulo(double largura, double altura) {
        this.largura = largura;
        this.altura = altura;
    }

    @Override
    public double calcularArea() {
        return largura * altura;
    }

    @Override
    public double calcularPerimetro() {
        return 2 * (largura + altura);
    }
}


public class Circulo implements FormaGeometrica {
    private double raio;

    public Circulo(double raio) {
        this.raio = raio;
    }
```

```
    @Override
    public double calcularArea() {
        return Math.PI * raio * raio;
    }

    @Override
    public double calcularPerimetro() {
        return 2 * Math.PI * raio;
    }
}


public class Main {
    public static void main(String[] args) {
        FormaGeometrica retangulo = new Retangulo(5, 10);
        System.out.println("Retângulo:");
        System.out.println("Área: " + retangulo.calcularArea());
        System.out.println("Perímetro: " + retangulo.calcularPerimetro());

        System.out.println();


        FormaGeometrica circulo = new Circulo(7);
        System.out.println("Círculo:");
        System.out.println("Área: " + circulo.calcularArea());
        System.out.println("Perímetro: " + circulo.calcularPerimetro());
    }
}
```

Exercício 4: No material da “Aula 07 – Relacionamento entre Classes”, no tópico “Alta Coesão”, nas páginas 23 e 24, implemente a classe para testes das classes listadas nessas páginas.

```
import java.util.ArrayList;
import java.util.List;


class Produto {
    private String nome;
    private double preco;

    public Produto(String nome, double preco) {
        this.nome = nome;
        this.preco = preco;
    }

    public double getPreco() {
        return preco;
    }

    @Override
```

```
    public String toString() {
        return nome + " - R$ " + preco;
    }
}


class Pedido {
    private List<Produto> produtos;

    public Pedido() {
        produtos = new ArrayList<>();
    }

    public void adicionarProduto(Produto produto) {
        produtos.add(produto);
    }

    public double calcularTotal() {
        double total = 0;
        for (Produto produto : produtos) {
            total += produto.getPreco();
        }
        return total;
    }

    public List<Produto> getProdutos() {
        return produtos;
    }
}


class Cliente {
    private String nome;

    public Cliente(String nome) {
        this.nome = nome;
    }

    public String getNome() {
        return nome;
    }
}


class GerenciadorDePedidos {
    public void finalizarPedido(Cliente cliente, Pedido pedido) {
        System.out.println("Pedido do cliente: " + cliente.getNome());
        System.out.println("Produtos no pedido:");
        for (Produto produto : pedido.getProdutos()) {
            System.out.println(produto);
        }
        System.out.println("Total do pedido: R$ " + pedido.calcularTotal());
    }
}
```

```
public class Main {
    public static void main(String[] args) {

        Produto p1 = new Produto("Livro", 20.0);
        Produto p2 = new Produto("Caneta", 5.0);


        Pedido pedido = new Pedido();
        pedido.adicionarProduto(p1);
        pedido.adicionarProduto(p2);


        Cliente cliente = new Cliente("João");


        GerenciadorDePedidos gerenciador = new GerenciadorDePedidos();
        gerenciador.finalizarPedido(cliente, pedido);
    }
}
```

Exercício 5: No material da “Aula 07 – Relacionamento entre Classes”, no tópico “Baixo Acoplamento”, na página 29, implemente a classe para testes das classes listadas nessa página.

```
interface Notificacao {
    void enviar(String mensagem);
}


class EmailNotificacao implements Notificacao {
    @Override
    public void enviar(String mensagem) {
        System.out.println("Enviando e-mail com a mensagem: " + mensagem);
    }
}


class SMSNotificacao implements Notificacao {
    @Override
    public void enviar(String mensagem) {
        System.out.println("Enviando SMS com a mensagem: " + mensagem);
    }
}


class Notificador {
    private Notificacao notificacao;
```

```
    public Notificador(Notificacao notificacao) {
        this.notificacao = notificacao;
    }

    public void setNotificacao(Notificacao notificacao) {
        this.notificacao = notificacao;
    }

    public void notificar(String mensagem) {
        notificacao.enviar(mensagem);
    }
}


public class Main {
    public static void main(String[] args) {

        Notificador notificadorEmail = new Notificador(new EmailNotificacao());
        notificadorEmail.notificar("Bem-vindo ao sistema!");

        System.out.println(); // linha em branco para separar os testes


        Notificador notificadorSMS = new Notificador(new SMSNotificacao());
        notificadorSMS.notificar("Sua conta foi atualizada.");
    }
}
```

Exercício 6: No material da “Aula 07 – Relacionamento entre Classes”, no tópico “Herança Simples”, na página 34, implemente a classe para testes das classes listadas nessa página.

```
public class Pessoa {
    private String nome;
    private int idade;
    private String endereco;
    private String telefone;


    public String getNome() {
        return nome;
    }

    public void setNome(String nome) {
        this.nome = nome;
    }


}
```

```
import org.junit.Test;
import static org.junit.Assert.*;

public class TestePessoa {

    @Test
    public void testSetNome() {
        Pessoa pessoa = new Pessoa();
        pessoa.setNome("João da Silva");
        assertEquals("João da Silva", pessoa.getNome());
    }

    @Test
    public void testSetIdade() {
        Pessoa pessoa = new Pessoa();
        pessoa.setIdade(30);
        assertEquals(30, pessoa.getIdade());
    }

    @Test
    public void testSetEndereco() {
        Pessoa pessoa = new Pessoa();
        pessoa.setEndereco("Rua das Flores, 123");
        assertEquals("Rua das Flores, 123", pessoa.getEndereco());
    }

    @Test
    public void testSetTelefone() {
        Pessoa pessoa = new Pessoa();
        pessoa.setTelefone("(11) 98765-4321");
        assertEquals("(11) 98765-4321", pessoa.getTelefone());
    }

    @Test
    public void testSetNomeComNull() {
        Pessoa pessoa = new Pessoa();
        assertThrows(IllegalArgumentException.class, () -> pessoa.setNome(null));
    }

}
```