DISCIPLINA: **COMPLIANCE & QUALITY ASSURANCE**

AULA **2 – EXEMPLIFICAÇÃO DE COMO FERRAMENTAS E PROCESSOS IMPACTAM A QUALIDADE– JUNIT**

ESTUDO DE CASO

PROFESSOR:
RENATO JARDIM PARDUCCI

PROFRENATO.PARDUCCI@FIAP.COM.BR

Renato Parducci - YouTube

Agora, crie a Classe descrita ao lado e a JUNIT para testar todos os métodos da Classe.

Faça os testes para criar um objeto e instanciá-lo e depois testar a recuperação de dados.

```
public class  Produto{

    private double peso;
    private double altura;

    public double getPeso() {
      return peso;
    }

    public void setPeso(double peso) {
      this.peso = peso;
    }

    public double getAltura() {
      return altura;
    }

    public void setAltura(double altura) {
      this.altura = altura;
    }

}
```

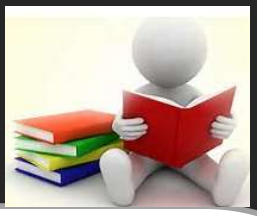
Classe JAVA ...

```
public class Produto {

        private double peso;
        private double altura;

        public Produto() {

        }

        public Produto(double peso, double altura) {
                this.peso=peso;
                this.altura=altura;
        }

        public double getPeso() {
                return peso;
        }

        public void setPeso(double peso) {
                this.peso = peso;
        }

        public double getAltura() {
                return altura;
        }

        public void setAltura(double altura) {
                this.altura = altura;
        }

}
```

Os casos de teste...

```
import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.Test;

class ProdutoTeste {

	@Test
	void testGetterPeso() {

		Produto pr = new Produto(75, 180);
		double resultadoEsperado = 75;
		double resultadoReal = pr.getPeso();

		assertEquals(resultadoEsperado, resultadoReal);
	}

	@Test
	void testGetterAltura() {

		Produto pr = new Produto(75, 180);
		double resultadoEsperado = 180;
		double resultadoReal = pr.getAltura();

		assertEquals(resultadoEsperado, resultadoReal);
	}

	@Test
	void testSetterPeso() {

		Produto pr = new Produto();
		//set eh void

		pr.setPeso(75);

		double resultadoEsperado = 75;
		double resultadoReal = pr.getPeso();

		assertEquals(resultadoEsperado, resultadoReal);

	}

	@Test
	void testSetterAltura() {

		Produto pr = new Produto();
		//set eh void

		pr.setAltura(180);

		double resultadoEsperado = 180;
		double resultadoReal = pr.getAltura();

		assertEquals(resultadoEsperado, resultadoReal);

	}
}
```

ESTUDO DE CASO SIMULADO



Você está em um projeto de um software e precisa criar uma função que colha 3 digitações de números inteiros, compare os três e diga qual o maior e qual o menor número digitado.

1º) Crie a Classe JAVA e o Método de Captura de Digitação de um número, mais o Método exibir o Maior e outro para exibir o Menor número entre três digitados.

2º) Crie a Classe de Testes e seus Métodos, com JUNIT.

3º) Crie depois a Suíte de teste.
Crie a classe executora dos testes.

4º) Depois que tudo estiver funcionando, transfira a Classe e as JUNITs para o projeto que foi aplicado no treinamento (Calculadora Digital) e ajuste a Test Suite Class e a Test Runner para acomodar essa nova classe e seus testes.

COMPLIANCE & QUALITY ASSURANCE

Classe JAVA com o método de coleta de número para comparação:

```
import java.util.Scanner;

public class MinMax {
    public int capturaNro () {
        Scanner sc = new Scanner(System.in);
        int n1;

        System.out.print("Entre com o número  inteiro: ");
        n1 = sc.nextInt();
       return n1;
    }
}
```

COMPLIANCE & QUALITY ASSURANCE

JUNIT de teste da digitação de números a comparar:

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

import org.junit.Test;

public class MinMaxTest {

        @Test
        public void testDigitacao() {
                int nroDigitado = 0;
                int valEsperado = 1;
                MinMax comparadora = new MinMax();
                nroDigitado = comparadora.capturaNro();
                assertEquals(nroDigitado, valEsperado);
                }
}
```

Classe JAVA com o método para encontrar e retornar o menor número entre três:

```
import java.util.Scanner;

public class MinMax {
 public int retornaMin (int n1, int n2, int n3) {
      if (n1 > n2) {
         if (n1 > n3) {
            if (n2 < n3) {
               System.out.println("O menor numero: " + n2);
               return n2;
            } else {
               System.out.println("O menor numero: " + n3);
               return n3;
            }

         } else {
            if (n1 < n2) {
               System.out.println("O menor numero: " + n1);
               return n1;
            } else {
               System.out.println("O menor numero: " + n2);
               return n2;
            }
         }
      } else {
         if (n2 > n3) {
            if (n1 < n3) {
               System.out.println("O menor numero: " + n1);
               return n1;
            } else {
               System.out.println("O menor numero: " + n3);
               return n3;
            }
         } else {
            if (n1 < n2) {
               System.out.println("O menor numero: " + n1);
               return n1;
            } else {
               System.out.println("O menor numero: " + n2);
               return n3;
            }
         }
      }

   }}
```

COMPLIANCE & QUALITY ASSURANCE

JUNIT incluindo o teste para identificar o menor número:

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

import org.junit.Test;

public class MinMaxTest {

		@Test
		public void testDigitacao() {
				int nroDigitado = 0;
				int valEsperado = 1;
				MinMax comparadora = new MinMax();
				nroDigitado = comparadora.capturaNro();
				assertEquals(nroDigitado, valEsperado);
				}

		@Test
		public void testMin() {
				int nroDigitado1 = 0;
				int nroDigitado2 = 1;
				int nroDigitado3= 2;
				int valEsperado = 0;
				MinMax apuradoraMin = new MinMax();
				int nroMinEncontrado = apuradoraMin.retornaMin(nroDigitado1, nroDigitado2, nroDigitado3);
				assertEquals(nroMinEncontrado, valEsperado);
				}

}
```

COMPLIANCE & QUALITY ASSURANCE

TEST SUITE da digitação de número e exibição do menor

```
import org.junit.runner.RunWith;
import org.junit.runners.Suite;
import org.junit.runners.Suite.SuiteClasses;

@RunWith(Suite.class)
@SuiteClasses({ MinMaxTest.class })
public class SuiteTestMin {

}
```

COMPLIANCE & QUALITY ASSURANCE

Classe de execução de teste (TEST RUNNER) da digitação de número e exibição do menor

```
import org.junit.runner.JUnitCore;
import org.junit.runner.Result;
import org.junit.runner.notification.Failure;

public class ExecutoraTestes {
  public static void main(String[] args) {
    Result resultado = JUnitCore.runClasses(SuiteTestMin.class);

    for (Failure failure : resultado.getFailures()) {
      System.out.println(failure.toString());
    }

    System.out.println(resultado.wasSuccessful());
  }

}
```