

DISCIPLINA: PROJETO DE SISTEMAS APLICADO AS MELHORES PRÁTICAS EM QUALIDADE DE SOFTWARE E GOVERNANÇA DE TI

Lista de exercícios e atividades complementares

1 – INTRODUÇÃO ÀS BOAS PRÁTICAS DE DESENVOLVIMENTO DE SISTEMAS

PROFESSOR:
RENATO JARDIM PARDUCCI

PROFRENATO.PARDUCCI@FIAP.COM.BR

[Renato Parducci - YouTube](#)

**ATIVIDADES
COMPLEMENTARES**

Tente reproduzir os seguintes passos para operar o GIT HUB:

- 1º) Selecione um conjunto de arquivos nos diretórios do seu computador para administrar com GIT HUB. Busque um arquivo fonte SQL, um JAVA e um .DOC.
- 2º) Crie um projeto no GIT HUB com pastas separadas para fontes de aplicação e para documentos auxiliares.
- 3º) Suba o .DOC na área de documentos auxiliares e os outros arquivos na área de fontes de aplicação, usando uma Branch de manutenção
- 4º) Abra o arquivo .SQL, edite, aprobe e republique na Master. Repita o processo com o .JAVA.
- 5º) Faça download do .DOC para manutenção, usando uma Branch. Edite o arquivo usando o MS-Word e devolva ao GIT por upload.
- 6º) Confira o histórico de versões dos arquivos.
- 7º) Recupere a versão anterior do .SQL.

Tire suas dúvidas com o professor.

Tente reproduzir os seguintes passos para operar o GIT HUB:

Crie testes com JUNIT para um programa de aplicação que você. está desenvolvendo na disciplina de programação JAVA.

Tire suas dúvidas com o professor.

EXERCÍCIOS RESOLVIDOS

1. Elabore casos de teste com JUNIT para a Classe/Método a seguir:

```
Calculo.java - Eclipse Platform
Source Refactor Navigate Search Project Run Window Help

package processos;

public class Calculo {
    public static float ExecutaCalculo(float Valor1, float Valor2)
    {
        float Soma = Valor1 + Valor2;
        return Soma;
    }
}
```

2. Elabore casos de teste com JUNIT para comparar duas variáveis do tipo String com as seguintes combinações:

-Situação 1: variável resultadoEsperado tem o valor “Registro salvo com sucesso” e resultadoObtido tem o mesmo valor

-Situação 2: variável resultadoEsperado tem o valor “Registro salvo com sucesso” e resultadoObtido tem o valor “Funcionário já existe”

3. Elabore casos de teste com JUNIT para a Classe a seguir e seus Métodos:

```
public class DilemaDoPrisioneiro {  
  
    private int PENA_INOCENCIA = 0;  
    private int PENA_CONDENACAO_MUTUA = 5;  
    private int PENA_CONDENACAO_INDIVIDUAL = 10;  
    private int PENA_CONDENACAO_CUPLICES = 1;  
  
    public enum Resposta {  
        NEGACAO, DELACAO  
    }  
  
    public int calculaPena(Resposta respostaPrisioneiroA, Resposta respostaPrisioneiroB)  
  
        if (respostaPrisioneiroA == Resposta.DELACAO) {  
  
            if (respostaPrisioneiroB == Resposta.DELACAO) {  
                return PENA_CONDENACAO_MUTUA;  
            } else {  
                return PENA_INOCENCIA;  
            }  
  
        } else {  
  
            if (respostaPrisioneiroB == Resposta.DELACAO) {  
                return PENA_CONDENACAO_INDIVIDUAL;  
            } else {  
                return PENA_CONDENACAO_CUPLICES;  
            }  
  
        }  
  
    }  
  
}
```


LISTA DE EXERCÍCIOS DE FIXAÇÃO

Exercícios propostos

4. Elabore casos de teste com JUNIT para a Classe/Método a seguir:

```
public class Hello {  
    public static void main(String[] args) {  
        System.out.println("Hello, world!");  
    }  
}
```

5. Elabore casos de teste com JUNIT para a Classe/Método a seguir:

```
public class MinMax1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n1, n2, n3;

        System.out.print("Entre com o primeiro inteiro: ");
        n1 = sc.nextInt();
        System.out.print("Entre com o segundo inteiro: ");
        n2 = sc.nextInt();
        System.out.print("Entre com o terceiro inteiro: ");
        n3 = sc.nextInt();
        if (n1 > n2) {
            if (n1 > n3) {
                if (n2 < n3) {
                    System.out.println("O menor numero eh: " + n2);
                } else {
                    System.out.println("O menor numero eh: " + n3);
                }
                System.out.println("O maior numero eh: " + n1);
            } else {
                if (n1 < n2) {
                    System.out.println("O menor numero eh: " + n1);
                } else {
                    System.out.println("O menor numero eh: " + n2);
                }
                System.out.println("O maior numero eh: " + n3);
            }
        } else {
            if (n2 > n3) {
                if (n1 < n3) {
                    System.out.println("O menor numero eh: " + n1);
                } else {
                    System.out.println("O menor numero eh: " + n3);
                }
                System.out.println("O maior numero eh: " + n2);
            } else {
                if (n1 < n2) {
                    System.out.println("O menor numero eh: " + n1);
                } else {
                    System.out.println("O menor numero eh: " + n2);
                }
                System.out.println("O maior numero eh: " + n3);
            }
        }
    }
}
```

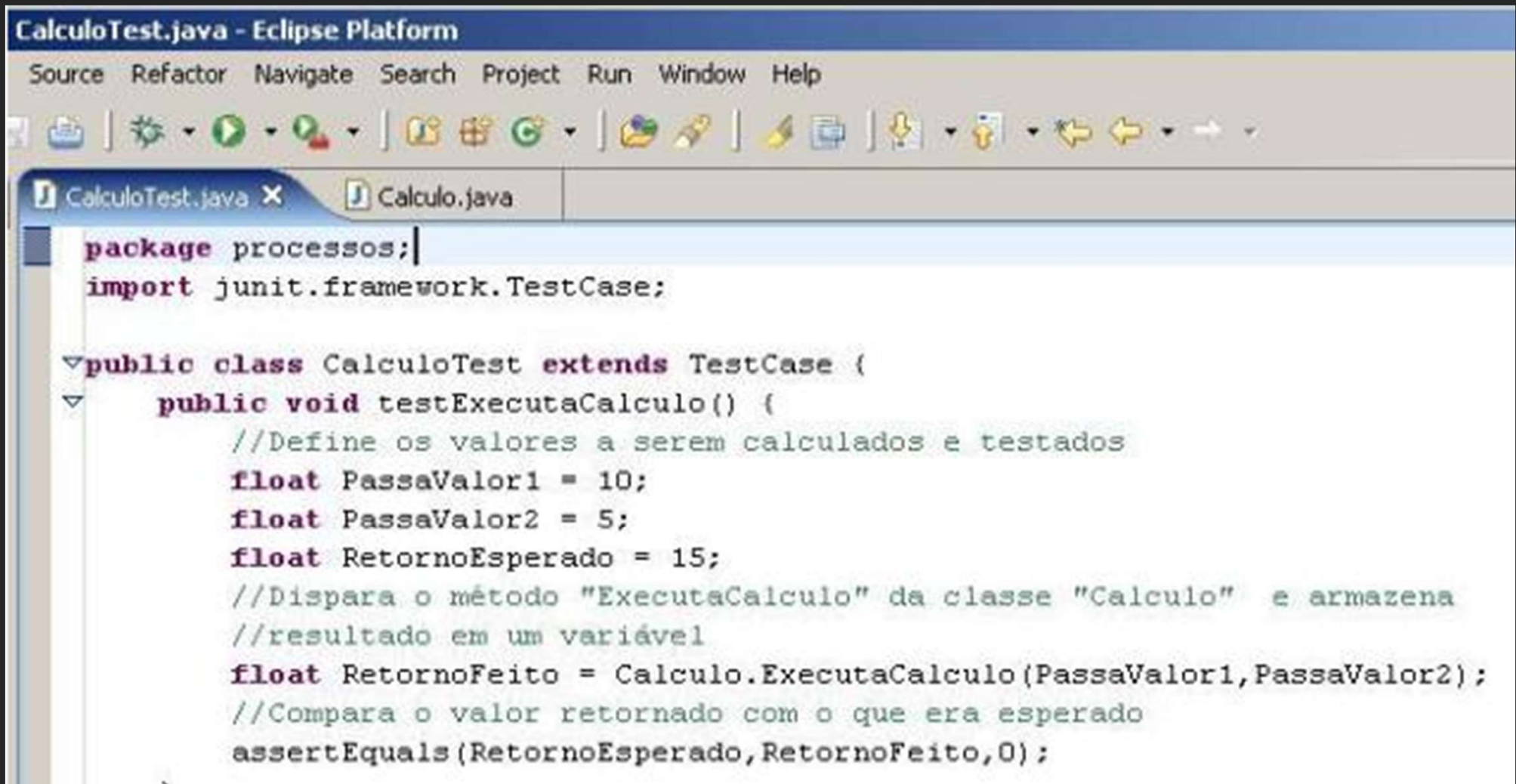
6. Elabore casos de teste com JUNIT para a Classe/Método a seguir:

```
public class Conta {  
  
    private Double saldo;  
  
    public void setSaldo(Double saldo) {  
        this.saldo = saldo;  
    }  
  
    public Double getSaldo() {  
        return saldo;  
    }  
  
    public void depositar(Double valor){  
        saldo += valor;  
    }  
  
    public void verificaSaldo(){  
        System.out.println("Valor do Saldo: "+getSaldo());  
    }  
}
```

LISTA DE EXERCÍCIOS DE FIXAÇÃO

Solução do Exercícios Resolvidos

1. Solução:



The screenshot shows the Eclipse IDE interface with the file 'CalculoTest.java' open. The code is as follows:

```

package processos;
import junit.framework.TestCase;

public class CalculoTest extends TestCase {
    public void testExecutaCalculo() {
        //Define os valores a serem calculados e testados
        float PassaValor1 = 10;
        float PassaValor2 = 5;
        float RetornoEsperado = 15;
        //Dispara o método "ExecutaCalculo" da classe "Calculo" e armazena
        //resultado em um variável
        float RetornoFeito = Calculo.ExecutaCalculo(PassaValor1,PassaValor2);
        //Compara o valor retornado com o que era esperado
        assertEquals(RetornoEsperado,RetornoFeito,0);
    }
}
    
```

2. Solução:

ExemploAssertEquals.java

```
1  import static org.junit.Assert.*;
2  import org.junit.Test;
3
4  public class AssertEqualsTest {
5
6      @Test
7      public void testeIgualdade_Sucesso() {
8          String resultadoEsperado = "Registro salvo com sucesso!";
9          String resultadoObtido = "Registro salvo com sucesso!";
10
11          assertEquals(resultadoEsperado, resultadoObtido);
12      }
13
14      @Test
15      public void testeIgualdade_Falha() {
16          String resultadoEsperado = "Registro salvo com sucesso!";
17          String resultadoObtido = "Funcionário já existe!";
18
19          assertEquals(resultadoEsperado, resultadoObtido);
```


3. Solução:

```
@Test
public void testCenario1() {

    Resposta respostaSuspeitoA = Resposta.DELACAO;
    Resposta respostaSuspeitoB = Resposta.DELACAO;

    DilemaDoPrisoneiro dp = new DilemaDoPrisoneiro();
    Assert.assertNotNull(dp);

    int penaSuspeitoA = dp.calculaPena(respostaSuspeitoA, respostaSuspeitoB);
    int penaSuspeitoB = dp.calculaPena(respostaSuspeitoB, respostaSuspeitoA);

    Assert.assertEquals(5, penaSuspeitoA);
    Assert.assertEquals(5, penaSuspeitoB);

}
```