PRO 5.3

No he podido conseguir como poder realizar el ejercicio bien, ya que aunque matriculado pero voy a anular mi matricula en Programación y la deje de lado. No he podido manejar bien la excepción y alguna otra cosa que me da error.

import java.util.Arrays;

import java.util.Collection;

import org.junit.Test;

import static org.junit.Assert.\*;

import org.junit.runner.RunWith;

import org.junit.runners.Parameterized;

@RunWith(Parameterized.class)

public class EmpleadoBRBrutoTest {

private String tipoEmpleado;

private float ventasMes;

private float horasExtra;

private float calculo;

private TipoEmpleado tipo;

public EmpleadoBRBrutoTest(String tipoEmpleado, float ventasMes, float horasExtra, float calculo) {

this.tipoEmpleado = tipoEmpleado;

this.ventasMes = ventasMes;

this.horasExtra = horasExtra;

this.calculo = calculo;

}

@Parameterized.Parameters

public static Collection<Object[]> salarios() {

return Arrays.asList(new Object[][]{

{TipoEmpleado.vendedor, 2000.00, 8.00, 1360.00},

{TipoEmpleado.vendedor, 1500.00, 3.00, 1260.00},

{TipoEmpleado.vendedor, 1499.99, 0, 1100.00},

{TipoEmpleado.encargado, 1250.00, 8.00, 1760.00},

{TipoEmpleado.encargado, 1000.00, 0, 1600.00},

{TipoEmpleado.encargado, 999.99, 3.00, 1560.00},

{TipoEmpleado.encargado, 500.00, 0, 1500.00},

{TipoEmpleado.encargado, 0, 8.00, 1600.00},

{null, 1500.00, 8.00, BRException.class},});

}

/\*\*

\* Test of calculaSalarioBruto method, of class EmpleadoBR.

\*/

@Test

public void testCalculaSalarioBruto() throws Exception {

EmpleadoBR instance = new EmpleadoBR(this.tipoEmpleado, this.ventasMes, this.horasExtra);

float ventasMes = 0.00F;

float horasExtra = 0.00F;

float expResult = this.calculo;

float result = instance.calculaSalarioBruto((TipoEmpleado) tipo, ventasMes, horasExtra);

assertEquals(this.calculo, result, 0);

}

}

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

import java.util.Arrays;

import java.util.Collection;

import org.junit.Test;

import static org.junit.Assert.\*;

import org.junit.runner.RunWith;

import org.junit.runners.Parameterized;

@RunWith(Parameterized.class)

public class EmpleadoBRNetoTest {

private float salarioBruto;

private float calculo;

public EmpleadoBRNetoTest(float salarioBruto) {

this.salarioBruto = salarioBruto;

this.calculo = calculo;

}

@Parameterized.Parameters

public static Collection<Object[]> salarios() {

return Arrays.asList(new Object[][]{

{2000, 1640},

{1500, 1230},

{1499.99, 1259.9916},

{1250, 1050},

{1000, 840},

{999.99, 999.99},

{500, 500},

{0, 0},});

}

/\*\*

\* Test of calculaSalarioNeto method, of class EmpleadoBR.

\*/

@Test

public void testCalculaSalarioNeto() throws Exception {

EmpleadoBR instance = new EmpleadoBR(this.salarioBruto);

float salarioBruto = 0.0F;

float expResult = 0.0F;

float result = instance.calculaSalarioNeto(salarioBruto);

assertEquals(expResult, result, 0);

}

}

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente con confianza media

TestSuite

import org.junit.After;

import org.junit.AfterClass;

import org.junit.Before;

import org.junit.BeforeClass;

import org.junit.runner.RunWith;

import org.junit.runners.Suite;

@RunWith(Suite.class)

@Suite.SuiteClasses({BRExceptionTest.class, EmpleadoBRBrutoTest.class, EmpleadoBRNetoTest.class, TipoEmpleadoTest.class})

public class EmpleadoBRsuite {

@BeforeClass

public static void setUpClass() throws Exception {

}

@AfterClass

public static void tearDownClass() throws Exception {

}

@Before

public void setUp() throws Exception {

}

@After

public void tearDown() throws Exception {

}

}

Texto

Descripción generada automáticamente