Davi Athanazio Quental – RM85868

***Governança e Melhores Práticas em Ti***

*CheckPoint 01 – JUNIT*

***Código Fonte:***

import static org.junit.Assert.\*;

import org.junit.Test;

public class CodigoPessoaFisicaTeste {

// Neste teste é informado um número de CPF correto para ser validado, o retorno esperado é um boolean TRUE

*@Test*

public void testaNumeroCpfCorreto() {

CodigoPessoaFisica validador = new CodigoPessoaFisica();

boolean retornoReal = validador.validaCPF("069.257.220-10");

*assertTrue*(retornoReal);

}

// Neste teste é informado um número de CPF incorreto para ser validado, o retorno esperado é um boolean FALSE

*@Test*

public void testaNumeroCpfErrado() {

CodigoPessoaFisica validador = new CodigoPessoaFisica();

boolean retornoReal = validador.validaCPF("111.444.777-05");

*assertFalse*(retornoReal);

}

// Neste teste é informado um número de CPF inválido, que apesar de correto, o retorno esperado é um boolean FALSE

*@Test*

public void testaNumeroCpfInvalido() {

CodigoPessoaFisica validador = new CodigoPessoaFisica();

boolean retornoReal = validador.validaCPF("000.000.000-00");

*assertFalse*(retornoReal);

}

// Neste teste é informado um número de CPF que possuí mais de 11 números, o retorno esperado é um boolean FALSE

*@Test*

public void testaNumeroCpfMaior() {

CodigoPessoaFisica validador = new CodigoPessoaFisica();

boolean retornoReal = validador.validaCPF("101.010.101-010");

*assertFalse*(retornoReal);

}

// Nesse teste é informado um CPF qualquer com "." e "-", o retorno esperado é o mesmo CPF sem os caracteres especiais

*@Test*

public void testaRemoverCaracteresEspeciais() {

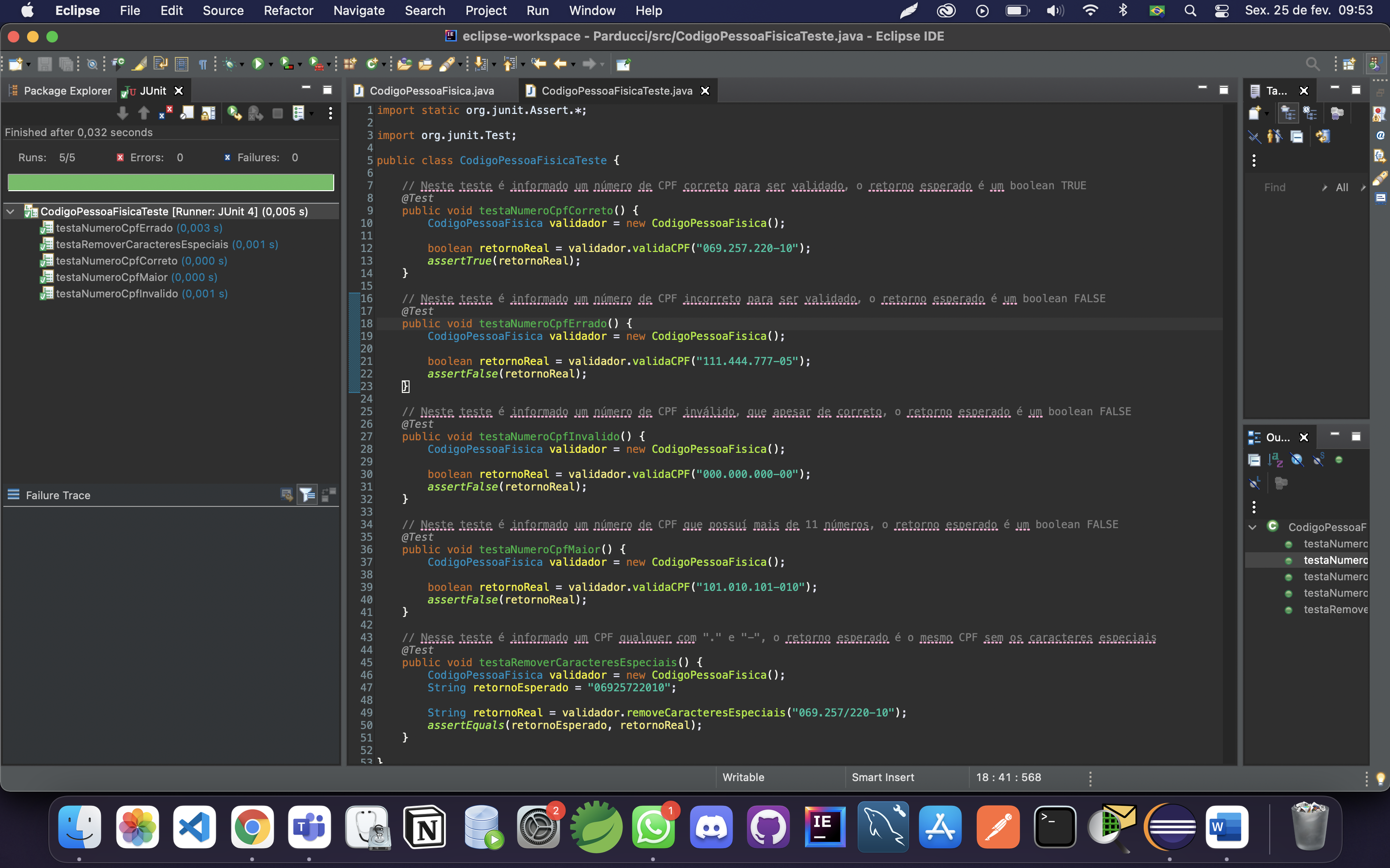
CodigoPessoaFisica validador = new CodigoPessoaFisica();

String retornoEsperado = "06925722010";

String retornoReal = validador.removeCaracteresEspeciais("069.257/220-10");

*assertEquals*(retornoEsperado, retornoReal);

***Print do JUnit e Código:***



}

}