Caue Adamo da Cruz

RM: 83645

import static org.junit.Assert.\*;

import org.junit.Test;

public class JulgamentoPrisioneiroTest {

@Test

public void test() {

fail("Not yet implemented");

}

}

Implementacao da classe de teste JUNIT

Falta de declaração/criação da variável RESPOSTA.

import org.junit.Assert;

import org.junit.Test;

public class JulgamentoPrisioneiroTest {

*@Test*

public void test() {

Resposta respostaSuspeitoA = Resposta.DELACAO;

Resposta respostaSuspeitoB = Resposta.DELACAO;

JulgamentoPrisioneiro jp = new JulgamentoPrisioneiro();

Assert.*assertNotNull*(jp);

int penaSuspeitoA = jp.calculaPena(respostaSuspeitoA, respostaSuspeitoB);

int penaSuspeitoB = jp.calculaPena(respostaSuspeitoB, respostaSuspeitoA);

Assert.*assertEquals*(15, penaSuspeitoA);

Assert.*assertEquals*(15, penaSuspeitoB);

}

}

Criacao da variável resposta - ENUM e testes

Ainda apontando erro no teste de JUNIT

import org.junit.Assert;

import org.junit.Test;

import JulgamentoPrisioneiro.Resposta;

public class JulgamentoPrisioneiroTest {

*@Test*

public void test() {

Resposta respostaSuspeitoA = Resposta.DELACAO;

Resposta respostaSuspeitoB = Resposta.DELACAO;

JulgamentoPrisioneiro jp = new JulgamentoPrisioneiro();

Assert.*assertNotNull*(jp);

int penaSuspeitoA = jp.calculaPena(respostaSuspeitoA, respostaSuspeitoB);

int penaSuspeitoB = jp.calculaPena(respostaSuspeitoB, respostaSuspeitoA);

Assert.*assertEquals*(15, penaSuspeitoA);

Assert.*assertEquals*(15, penaSuspeitoB);

}

}

Criacao do ENUM Resposta como classe, apagar a variável na classe JulgamentoPrisioneiro e rodar teste com sucesso

public class JulgamentoPrisioneiro {

private int PENA\_INOCENCIA = 13;

private int PENA\_CONDENACAO\_MUTUA = 15;

private int PENA\_CONDENACAO\_INDIVIDUAL = 10;

private int PENA\_CONDENACAO\_CUMPLICES = 11;

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\_CUMPLICES;

}

}

}

}

public enum *Resposta* {

***DELACAO***, ***NEGACAO***;

}

import org.junit.Assert;

import org.junit.Test;

public class JulgamentoPrisioneiroTest {

*@Test*

public void test() {

*Resposta* respostaSuspeitoA = *Resposta*.***DELACAO***;

*Resposta* respostaSuspeitoB = *Resposta*.***DELACAO***;

JulgamentoPrisioneiro jp = new JulgamentoPrisioneiro();

Assert.*assertNotNull*(jp);

int penaSuspeitoA = jp.calculaPena(respostaSuspeitoA, respostaSuspeitoB);

int penaSuspeitoB = jp.calculaPena(respostaSuspeitoB, respostaSuspeitoA);

Assert.*assertEquals*(15, penaSuspeitoA);

Assert.*assertEquals*(15, penaSuspeitoB);

}

}

Inclusão de 4 testes para 4 tipos de cenários

import org.junit.Assert;

import org.junit.Test;

public class JulgamentoPrisioneiroTest {

@Test

public void test1() {

Resposta respostaSuspeitoA = Resposta.DELACAO;

Resposta respostaSuspeitoB = Resposta.DELACAO;

JulgamentoPrisioneiro jp = new JulgamentoPrisioneiro();

Assert.assertNotNull(jp);

int penaSuspeitoA = jp.calculaPena(respostaSuspeitoA, respostaSuspeitoB);

int penaSuspeitoB = jp.calculaPena(respostaSuspeitoB, respostaSuspeitoA);

Assert.assertEquals(15, penaSuspeitoA);

Assert.assertEquals(15, penaSuspeitoB);

}

@Test

public void test2() {

Resposta respostaSuspeitoA = Resposta.DELACAO;

Resposta respostaSuspeitoB = Resposta.NEGACAO;

JulgamentoPrisioneiro jp = new JulgamentoPrisioneiro();

Assert.assertNotNull(jp);

int penaSuspeitoA = jp.calculaPena(respostaSuspeitoA, respostaSuspeitoB);

int penaSuspeitoB = jp.calculaPena(respostaSuspeitoB, respostaSuspeitoA);

Assert.assertEquals(13, penaSuspeitoA);

Assert.assertEquals(10, penaSuspeitoB);

}

@Test

public void test3() {

Resposta respostaSuspeitoA = Resposta.NEGACAO;

Resposta respostaSuspeitoB = Resposta.NEGACAO;

JulgamentoPrisioneiro jp = new JulgamentoPrisioneiro();

Assert.assertNotNull(jp);

int penaSuspeitoA = jp.calculaPena(respostaSuspeitoA, respostaSuspeitoB);

int penaSuspeitoB = jp.calculaPena(respostaSuspeitoB, respostaSuspeitoA);

Assert.assertEquals(11, penaSuspeitoA);

Assert.assertEquals(11, penaSuspeitoB);

}

@Test

public void test4() {

Resposta respostaSuspeitoA = Resposta.NEGACAO;

Resposta respostaSuspeitoB = Resposta.DELACAO;

JulgamentoPrisioneiro jp = new JulgamentoPrisioneiro();

Assert.assertNotNull(jp);

int penaSuspeitoA = jp.calculaPena(respostaSuspeitoA, respostaSuspeitoB);

int penaSuspeitoB = jp.calculaPena(respostaSuspeitoB, respostaSuspeitoA);

Assert.assertEquals(10, penaSuspeitoA);

Assert.assertEquals(13, penaSuspeitoB);

}

}