**Classe**

**public** **class** JulgamentoPrisioneiro {

**private** **int** PENA\_INOCENCIA = 10;

**private** **int** PENA\_CONDENACAO\_MUTUA = 15;

**private** **int** PENA\_CONDENACAO\_INDIVIDUAL = 10;

**private** **int** PENA\_CONDENACAO\_CUMPLICES = 11;

**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\_CUMPLICES;

}

}

}

}

**JUNIT Teste**

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

**import** org.junit.jupiter.api.Test;

**class** JulgamentoPrisioneiroTeste **extends** JulgamentoPrisioneiro {

@Test

**public** **void** testCenario1() {

Resposta respostaPrisioneiroA = Resposta.***DELACAO***;

Resposta respostaPrisioneiroB = Resposta.***DELACAO***;

JulgamentoPrisioneiro jp = **new** JulgamentoPrisioneiro();

*assertNotNull*(jp);

**int** penaPrisioneiroA = jp.calculaPena(respostaPrisioneiroA, respostaPrisioneiroB);

**int** penaPrisioneiroB = jp.calculaPena(respostaPrisioneiroB, respostaPrisioneiroA);

*assertEquals*(15, penaPrisioneiroA);

*assertEquals*(15, penaPrisioneiroB);

;

}}

**TESTE - 1**

Resposta não é uma variável disponível ( não criada )

**CORREÇÃO - 1**

Criando a ENUM Resposta

**\*implementação de todos os métodos na classe de teste antes de realizar o segundo teste**

**CLASSE JUNIT TESTE**

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

**import** org.junit.Assert;

**import** org.junit.jupiter.api.Test;

**class** JulgamentoPrisioneiroTeste **extends** JulgamentoPrisioneiro {

@Test

**public** **void** testCenario1() {

Resposta respostaPrisioneiroA = Resposta.***DELACAO***;

Resposta respostaPrisioneiroB = Resposta.***DELACAO***;

JulgamentoPrisioneiro jp = **new** JulgamentoPrisioneiro();

Assert.*assertNotNull*(jp);

**int** penaPrisioneiroA = jp.calculaPena(respostaPrisioneiroA, respostaPrisioneiroB);

**int** penaPrisioneiroB = jp.calculaPena(respostaPrisioneiroB, respostaPrisioneiroA);

Assert.*assertEquals*(15, penaPrisioneiroA);

Assert.*assertEquals*(15, penaPrisioneiroB);

;

}

@Test

**public** **void** testCase2() {

Resposta respostaPrisioneiroA = Resposta.***NEGACAO***;

Resposta respostaPrisioneiroB = Resposta.***NEGACAO***;

JulgamentoPrisioneiro jp = **new** JulgamentoPrisioneiro();

Assert.*assertNotNull*(jp);

**int** penaPrisioneiroA = jp.calculaPena(respostaPrisioneiroA, respostaPrisioneiroB);

**int** penaPrisioneiroB = jp.calculaPena(respostaPrisioneiroB, respostaPrisioneiroA);

Assert.*assertEquals*(10, penaPrisioneiroA);

Assert.*assertEquals*(10, penaPrisioneiroB);

}

**public** **void** testCenario3() {

Resposta respostaPrisioneiroA = Resposta.***DELACAO***;

Resposta respostaPrisioneiroB = Resposta.***NEGACAO***;

JulgamentoPrisioneiro jp = **new** JulgamentoPrisioneiro();

Assert.*assertNotNull*(jp);

**int** penaPrisioneiroA = jp.calculaPena(respostaPrisioneiroA, respostaPrisioneiroB);

**int** penaPrisioneiroB = jp.calculaPena(respostaPrisioneiroB, respostaPrisioneiroA);

Assert.*assertEquals*(10, penaPrisioneiroA);

}

}

**CLASSE**

**public** **class** JulgamentoPrisioneiro {

**private** **int** PENA\_INOCENCIA = 10;

**private** **int** PENA\_CONDENACAO\_MUTUA = 15;

**private** **int** PENA\_CONDENACAO\_INDIVIDUAL = 10;

**private** **int** PENA\_CONDENACAO\_CUMPLICES = 11;

**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\_CUMPLICES;

}

}

}

}

**TESTE - 2**

**Erro no assert equals do Case2**

**CORREÇÃO - 2**

**Troca no if else dos retornos(pena cumplice > pena inocência)**

**CLASSE**

**public** **class** JulgamentoPrisioneiro {

**private** **int** PENA\_INOCENCIA = 10;

**private** **int** PENA\_CONDENACAO\_MUTUA = 15;

**private** **int** PENA\_CONDENACAO\_INDIVIDUAL = 10;

**private** **int** PENA\_CONDENACAO\_CUMPLICES = 11;

**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\_CONDENACAO\_CUMPLICES;

}

} **else** {

**if** (respostaPrisioneiroB == Resposta.***DELACAO***) {

**return** PENA\_CONDENACAO\_INDIVIDUAL;

} **else** {

**return** PENA\_INOCENCIA;

}

}

}

}

**CLASSE JUNIT TESTE**

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

**import** org.junit.Assert;

**import** org.junit.jupiter.api.Test;

**class** JulgamentoPrisioneiroTeste **extends** JulgamentoPrisioneiro {

@Test

**public** **void** testCenario1() {

Resposta respostaPrisioneiroA = Resposta.***DELACAO***;

Resposta respostaPrisioneiroB = Resposta.***DELACAO***;

JulgamentoPrisioneiro jp = **new** JulgamentoPrisioneiro();

Assert.*assertNotNull*(jp);

**int** penaPrisioneiroA = jp.calculaPena(respostaPrisioneiroA, respostaPrisioneiroB);

**int** penaPrisioneiroB = jp.calculaPena(respostaPrisioneiroB, respostaPrisioneiroA);

Assert.*assertEquals*(15, penaPrisioneiroA);

Assert.*assertEquals*(15, penaPrisioneiroB);

;

}

@Test

**public** **void** testCenario2() {

Resposta respostaPrisioneiroA = Resposta.***NEGACAO***;

Resposta respostaPrisioneiroB = Resposta.***NEGACAO***;

JulgamentoPrisioneiro jp = **new** JulgamentoPrisioneiro();

Assert.*assertNotNull*(jp);

**int** penaPrisioneiroA = jp.calculaPena(respostaPrisioneiroA, respostaPrisioneiroB);

**int** penaPrisioneiroB = jp.calculaPena(respostaPrisioneiroB, respostaPrisioneiroA);

Assert.*assertEquals*(10, penaPrisioneiroA);

Assert.*assertEquals*(10, penaPrisioneiroB);

}

**public** **void** testCenario3() {

Resposta respostaPrisioneiroA = Resposta.***DELACAO***;

Resposta respostaPrisioneiroB = Resposta.***NEGACAO***;

JulgamentoPrisioneiro jp = **new** JulgamentoPrisioneiro();

Assert.*assertNotNull*(jp);

**int** penaPrisioneiroA = jp.calculaPena(respostaPrisioneiroA, respostaPrisioneiroB);

**int** penaPrisioneiroB = jp.calculaPena(respostaPrisioneiroB, respostaPrisioneiroA);

Assert.*assertEquals*(10, penaPrisioneiroA);

}

}

**Teste 3-**

**Teste ok nenhum erro foi encontrado**