Gerder Silva RM 80582

Versão 1

**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** **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;

}

}

}

}

**class** TesteJUNIT {

@Test

**void** test() {

*fail*("Not yet implemented");

}

}

DESCRIÇÃO:

Criação da classe Java e da JUnit. Não existe a enum class Resposta.

Versão 2

**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;

**enum** Resposta{

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

}

**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;

}

}

}

}

class TesteJUNIT {

@Test

void test() {

fail("Not yet implemented");

}

}

Descrição:

Tendo em vista a falta da enum class Resposta na classe java, eu criei e já declarei os atributos da classe.

Versão 3

**class** TesteJUNIT {

@Test

**public** **void** CalculaPena\_Mutua() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***DELACAO***;

Resposta respB = Resposta.***DELACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(15, penaPrisioneiroA);

*assertEquals*(15, penaPrisioneiroB);

}

}

DESCRIÇÃO:

Criação do teste do primeiro cenário, Condenação Mútua.

Versão 4

**class** TesteJUNIT {

@Test

**public** **void** CalculaPena\_Mutua() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***DELACAO***;

Resposta respB = Resposta.***DELACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(15, penaPrisioneiroA);

*assertEquals*(15, penaPrisioneiroB);

}

**public** **void** CalculaPena\_Individual() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***DELACAO***;

Resposta respB = Resposta.***DELACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(10, penaPrisioneiroA);

*assertEquals*(10, penaPrisioneiroB);

}

}

DESRIÇÃO:

Criação do cenário 2, Condenação Individual.

Versão 5

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

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

//import JulgamentoPrisioneiro.Resposta;

**class** TesteJUNIT {

@Test

**public** **void** CalculaPena\_Mutua() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***DELACAO***;

Resposta respB = Resposta.***DELACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(15, penaPrisioneiroA);

*assertEquals*(15, penaPrisioneiroB);

}

**public** **void** CalculaPena\_Individual() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***DELACAO***;

Resposta respB = Resposta.***DELACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(10, penaPrisioneiroA);

*assertEquals*(10, penaPrisioneiroB);

}

**public** **void** CalculaPena\_Cumplices() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***DELACAO***;

Resposta respB = Resposta.***DELACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(11, penaPrisioneiroA);

*assertEquals*(11, penaPrisioneiroB);

}

}

DESCRIÇÃO:

Criação do cenário 3, Condenação Cumplices.

Versão 6

**public** **void** CalculaPena\_Inocencia() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***DELACAO***;

Resposta respB = Resposta.***DELACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(10, penaPrisioneiroA);

*assertEquals*(10, penaPrisioneiroB);

}

DESCRIÇÃO:

Criação do cenário 4, Inocência.

Versão 7

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

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

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

**private** **int** PENA\_CONDENACAO\_INDIVIDUAL = 0;

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

[...]

**public** **void** CalculaPena\_Inocencia() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***DELACAO***;

Resposta respB = Resposta.***DELACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(0, penaPrisioneiroA);

*assertEquals*(0, penaPrisioneiroB);

}

DESCRIÇÃO:

Correção do erro na “pena” Inocência. Antes calculava 10 anos para quem fosse inocentado. Agora já foi corrigido.

Versão 8 – FINAL

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

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

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

**private** **int** PENA\_CONDENACAO\_INDIVIDUAL = 0;

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

**public** **int** calculaPena(Resposta respA, Resposta respB) {

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

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

**return** PENA\_CONDENACAO\_MUTUA;

} **else** {

**return** PENA\_INOCENCIA;

}

} **else** {

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

**return** PENA\_CONDENACAO\_INDIVIDUAL;

} **else** {

**return** PENA\_CONDENACAO\_CUMPLICES;

}

}

}

}

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

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

//import JulgamentoPrisioneiro.Resposta;

**class** TesteJUNIT {

@Test

**public** **void** CalculaPena\_Mutua() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***DELACAO***;

Resposta respB = Resposta.***DELACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(15, penaPrisioneiroA);

*assertEquals*(15, penaPrisioneiroB);

}

**public** **void** CalculaPena\_Individual() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***DELACAO***;

Resposta respB = Resposta.***NEGACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(10, penaPrisioneiroA);

*assertEquals*(10, penaPrisioneiroB);

}

**public** **void** CalculaPena\_Cumplices() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***NEGACAO***;

Resposta respB = Resposta.***NEGACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(11, penaPrisioneiroA);

*assertEquals*(11, penaPrisioneiroB);

}

**public** **void** CalculaPena\_Inocencia() {

JulgamentoPrisioneiro prisioneiroA = **new** JulgamentoPrisioneiro();

JulgamentoPrisioneiro prisioneiroB = **new** JulgamentoPrisioneiro();

Resposta respA = Resposta.***NEGACAO***;

Resposta respB = Resposta.***DELACAO***;

**int** penaPrisioneiroA = prisioneiroA.calculaPena(respA, respB);

**int** penaPrisioneiroB = prisioneiroB.calculaPena(respB, respA);

*assertEquals*(0, penaPrisioneiroA);

*assertEquals*(0, penaPrisioneiroB);

}

}

DESCRIÇÃO:

NÃO HÁ MAIS ERROS NO CÓDIGO. Versão finalizada sem erros de compilação relatados.