



ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA



**DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
CARRERA DE INGENIERIA EN SISTEMAS E
INFORMÁTICA**

PROGRAMACIÓN MÓVIL

TEMA:

Unit Test

INTEGRANTES:

PASPUEL YÁNEZ MAYRA ALEXANDRA

QUISTANCHALA SUNTAXI KARLA DANIELA

VILLARRUEL ARCINIEGA MICHAEL ALEJANDRO

NRC:

6112

SANGOLQUÍ – ECUADOR

MAYO – SEPTIEMBRE 2020

Clase CalculadoraModeloTest

```
/*
 * ESPE - DCC - PROGRAMACIÓN MÓVIL
 * Sistema: Calculadora
 * Creado 30/05/2020
 * Modificado 16/06/2020
 *
 * Los contenidos de este archivo son propiedad privada y estan protegidos por
 * La licencia BSD
 *
 * Se puede utilizar, reproducir o copiar el contenido de este archivo.
 */
package com.example.calculadora;

import org.junit.Before;
import org.junit.Test;
import static org.junit.Assert.*;
/**
 * Clase que implementa el Unit Test
 *
 * @author Paspuel Mayra
 * @author Quistanchala Karla
 * @author Villarruel Michael
 */
public class CalculadoraModeloTest {

    private Operaciones nCalculadoraModelo;
    /**
     * Metodo setUp para inicializar el test
     */
    @Before
    public void setUp(){
        nCalculadoraModelo=new Operaciones();
    }
    /**
     * Metodo operacionNotNull para controlar Las operaciones
     */
    @Test
    public void operacionNotNull(){
        assertNotNull(nCalculadoraModelo);
    }

    /**
     * Metodo sumaEnteros para testear la función suma para números enteros
     */
    @Test
    public void sumaEnteros() {
        assertEquals(10,nCalculadoraModelo.suma(5,5),0.0);
    }

    /**
     * Metodo sumaDecimales para testear la función suma para números decimales
     */
    @Test
```

```

public void sumaDecimales() {
    assertEquals(8.4,nCalculadoraModelo.suma(5.2,3.2),0.0);
}

/**
 * Metodo sumaEnterosNegativos para testear la función suma para números
enteros negativos
 */
@Test
public void sumaEnterosNegativos() {
    assertEquals(-13,nCalculadoraModelo.suma(-8,-5),0.0);
}

/**
 * Metodo sumaDecimalesNegativos para testear la función suma para números
decimales negativos
 */
@Test
public void sumaDecimalesNegativos() {
    assertEquals(-10.1,nCalculadoraModelo.suma(-6.2,-3.9),0.0);
}

/**
 * Metodo restaEnteros para testear la función resta para números enteros
 */
@Test
public void restaEnteros() {
    assertEquals(4,nCalculadoraModelo.resta(23,19),0.0);
}

/**
 * Metodo restaDecimales para testear la función resta para números enteros
 */
@Test
public void restaDecimales() {
    assertEquals(22.1,nCalculadoraModelo.resta(23.3,1.2),0.0);
}

/**
 * Metodo restaEnterosNegativos para testear la función resta para números
enteros negativos
 */
@Test
public void restaEnterosNegativos() {
    assertEquals(-4,nCalculadoraModelo.resta(-23,-19),0.0);
}

/**
 * Metodo restaDecimalesNegativos para testear la función resta para números
enteros negativos
 */
@Test
public void restaDecimalesNegativos() {
    assertEquals(-22.1,nCalculadoraModelo.resta(-23.3,-1.2),0.0);
}

```

```

    /**
     * Metodo divisionEnteros para testear la función división para números
    enteros
     */
    @Test
    public void divisionEnteros() throws Exception {
        assertEquals(6,nCalculadoraModelo.division(36,6),0.0);
    }

    /**
     * Metodo divisionDecimales para testear la función división para números
    decimales
     */
    @Test
    public void divisionDecimales() throws Exception {
        assertEquals(10,nCalculadoraModelo.division(18,1.8),0.0);
    }

    /**
     * Metodo divisionEnterosNegativos para testear la función división para
    números enteros negativos
     */
    @Test
    public void divisionEnterosNegativos() throws Exception {
        assertEquals(6,nCalculadoraModelo.division(-30,-5),0.0);
    }

    /**
     * Metodo divisionDecimalesNegativos para testear la función división para
    números decimales
     */
    @Test
    public void divisionDecimalesNegativos() throws Exception {
        assertEquals(10,nCalculadoraModelo.division(-17,-1.7),0.0);
    }

    /**
     * Metodo multiplicacionEnteros para testear la función multiplicación para
    números enteros
     */
    @Test
    public void multiplicacionEnteros() {
        assertEquals(24,nCalculadoraModelo.multiplicacion(12,2),0.0);
    }

    /**
     * Metodo multiplicacionDecimales para testear la función multiplicación
    para números decimales
     */
    @Test
    public void multiplicacionDecimales() {
        assertEquals(9,nCalculadoraModelo.multiplicacion(6,1.5),0.0);
    }

```

```

/**
 * Metodo multiplicacionEnterosNegativos para testear la función
multiplicación para números enteros negativos
 */
@Test
public void multiplicacionEnterosNegativos() {
    assertEquals(20,nCalculadoraModelo.multiplicacion(-10,-2),0.0);
}

/**
 * Metodo multiplicacionDecimalesNegativos para testear la función
multiplicación para números decimales negativos
 */
@Test
public void multiplicacionDecimalesNegativos() {
    assertEquals(12,nCalculadoraModelo.multiplicacion(-8,-1.5),0.0);
}

/**
 * Metodo potenciaExponenteCero para testear la función potencia para
exponente 0
 */
@Test
public void potenciaExponenteCero() {
    assertEquals(1,nCalculadoraModelo.potencia(6,0),0.0);
}

/**
 * Metodo potenciaExponenteEnteroNegativo para testear la función potencia
para exponente negativo
 */
@Test
public void potenciaExponenteEnteroNegativo() {
    assertEquals(0.25,nCalculadoraModelo.potencia(0.5,2),0.0);
}

/**
 * Metodo potenciaExponenteEnteroPositivo para testear la función potencia
para exponente positivo
 */
@Test
public void potenciaExponenteEnteroPositivo() {
    assertEquals(32,nCalculadoraModelo.potencia(2,5),0.0);
}

/**
 * Metodo potenciaExponenteDecimal para testear la función potencia para
decimales
 */
@Test
public void potenciaExponenteDecimal() {
    assertEquals(2,nCalculadoraModelo.potencia(4,0.5),0.0);
}

/**

```

```

    * Metodo factorialEnteroPositivo para testear la función factorial con
    ingreso de entero positivo
    */
    @Test
    public void factorialEnteroPositivo() throws Exception {
        assertEquals(6,nCalculadoraModelo.factorial(3),0);
    }

    @Test
    public void raizNumeroEntero() throws Exception {
        assertEquals(4,nCalculadoraModelo.raiz(16),0.0);
    }

    @Test
    public void raizNumeroDecimal() throws Exception {
        assertEquals(0.5,nCalculadoraModelo.raiz(0.25),0.0);
    }

    /**
     * Metodo LogaritmoDecimal para testear la función Logaritmo con ingreso de
     decimales
     */
    @Test
    public void logaritmoDecimal() throws Exception {
        assertEquals(2.09829,nCalculadoraModelo.logaritmo(125.4),0);
    }

    /**
     * Metodo LogaritmoEntero para testear la función Logaritmo con ingreso de
     enteros
     */
    @Test
    public void logaritmoEntero() throws Exception {
        assertEquals(1.69897,nCalculadoraModelo.logaritmo(50),0);
    }

    /**
     * Metodo modDenominadorNegativo para testear la función mod con denominador
     negativo
     */
    @Test
    public void modDenominadorNegativo() throws Exception {
        assertEquals(-1,nCalculadoraModelo.mod(5,-3),0);
    }

    /**
     * Metodo modNumeradorNegativo para testear la función mod con denominador
     negativo
     */
    @Test
    public void modNumeradorNegativo() throws Exception {
        assertEquals(3,nCalculadoraModelo.mod(-5,8),0);
    }

    /**
     * Metodo SenoEntero para testear la función seno con ingreso de enteros

```

```

    */
@Test
public void senoEntero() throws Exception {
    assertEquals(0.86,nCalculadoraModelo.seno(60),0.01);
}

/**
 * Metodo CosenoEntero para testear La función coseno con ingreso de enteros
 */
@Test
public void cosenoEntero() throws Exception {
    assertEquals(-1,nCalculadoraModelo.coseno(180),0.01);
}

/**
 * Metodo Tangente para testear La función tangente con ingreso de enteros
 */
@Test
public void tangenteEntero() throws Exception {
    assertEquals(1,nCalculadoraModelo.tangente(45),0.01);
}

/**
 * Metodo Binarios para testear La función binario con ingreso de decimales
 */
@Test
public void binarioEntero() throws Exception {
    assertEquals("0011",nCalculadoraModelo.binario(3));
}

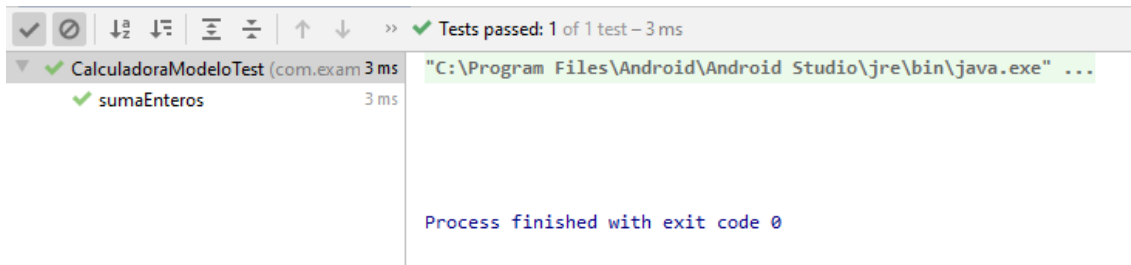
/**
 * Metodo Octal para testear La función octal con ingreso de decimales
 */
@Test
public void octalEntero() throws Exception {
    assertEquals("71",nCalculadoraModelo.octal(57));
}

/**
 * Metodo Hexadecimal para testear La función octal con ingreso de decimales
 */
@Test
public void hexadecimalEntero() throws Exception {
    assertEquals("3C",nCalculadoraModelo.hexadecimal(60));
}

/**
 * Metodo mR para testear La función de mMas, mMenos y mR
 */
@Test
public void mR() {
    nCalculadoraModelo.mMas(5);
    nCalculadoraModelo.mMenos(2.1);
    assertEquals(2.9,nCalculadoraModelo.mR(),0.0);
}
}

```

Test de suma de enteros



Tests passed: 1 of 1 test – 3 ms

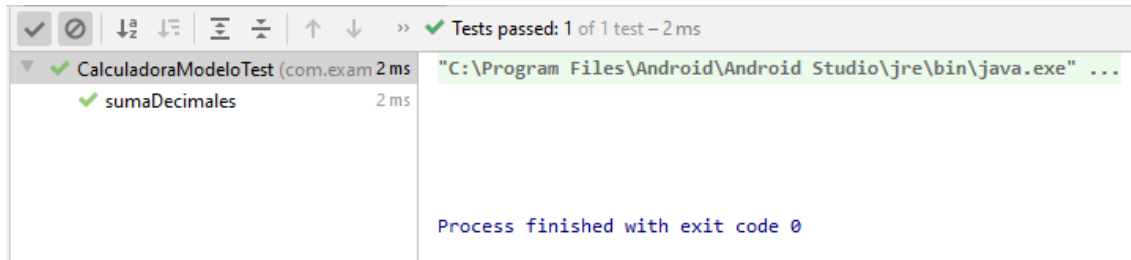
CalculadoraModeloTest (com.exam) 3 ms

✓ sumaEnteros 3 ms

"C:\Program Files\Android\Android Studio\jre\bin\java.exe" ...

Process finished with exit code 0

Test de suma de decimales



Tests passed: 1 of 1 test – 2 ms

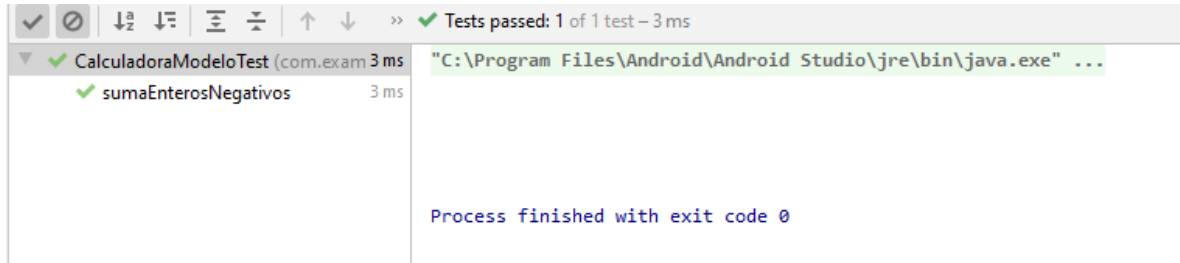
CalculadoraModeloTest (com.exam) 2 ms

✓ sumaDecimales 2 ms

"C:\Program Files\Android\Android Studio\jre\bin\java.exe" ...

Process finished with exit code 0

Test de suma de enteros negativos



Tests passed: 1 of 1 test – 3 ms

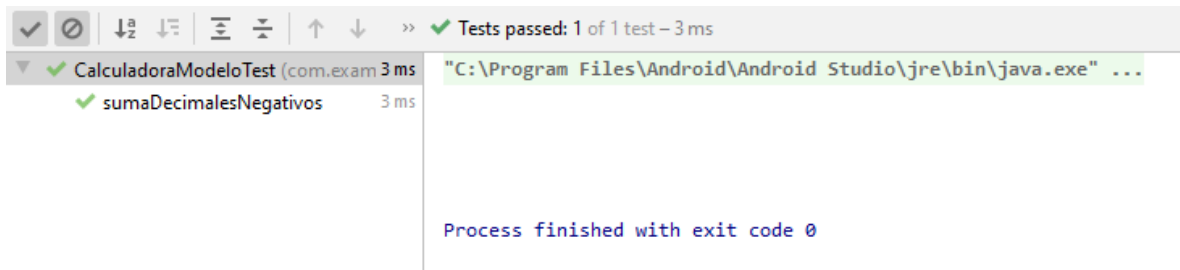
CalculadoraModeloTest (com.exam) 3 ms

✓ sumaEnterosNegativos 3 ms

"C:\Program Files\Android\Android Studio\jre\bin\java.exe" ...

Process finished with exit code 0

Test de suma de decimales negativos



Tests passed: 1 of 1 test – 3 ms

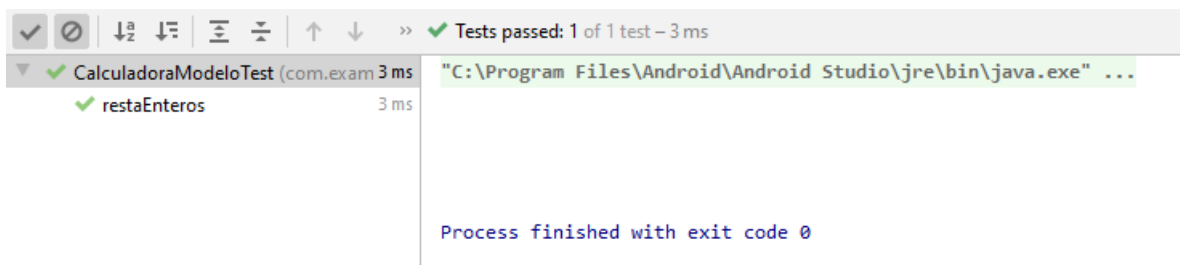
CalculadoraModeloTest (com.exam) 3 ms

✓ sumaDecimalesNegativos 3 ms

"C:\Program Files\Android\Android Studio\jre\bin\java.exe" ...

Process finished with exit code 0

Test de resta de enteros



Tests passed: 1 of 1 test – 3 ms

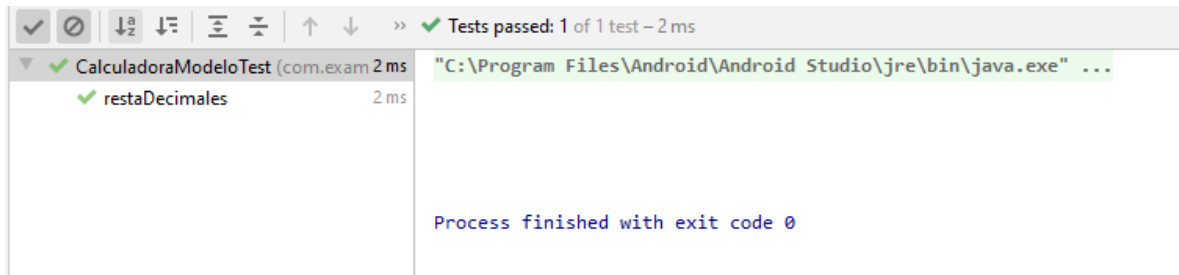
CalculadoraModeloTest (com.exam) 3 ms

✓ restaEnteros 3 ms

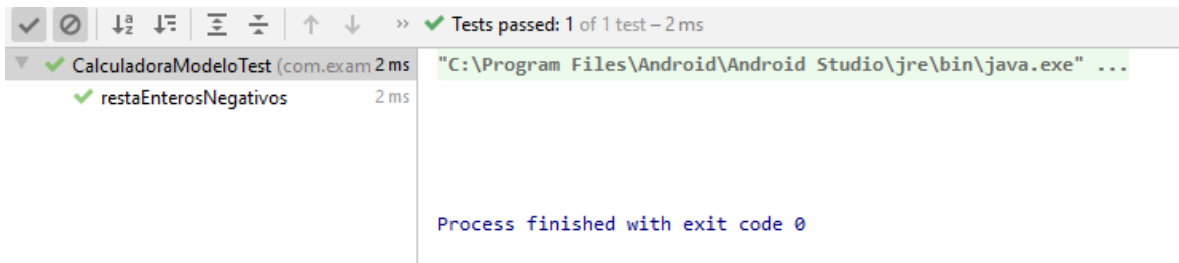
"C:\Program Files\Android\Android Studio\jre\bin\java.exe" ...

Process finished with exit code 0

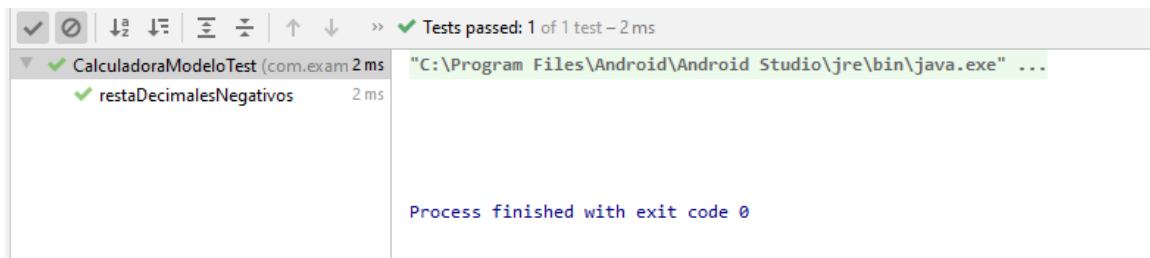
Test de resta de decimales



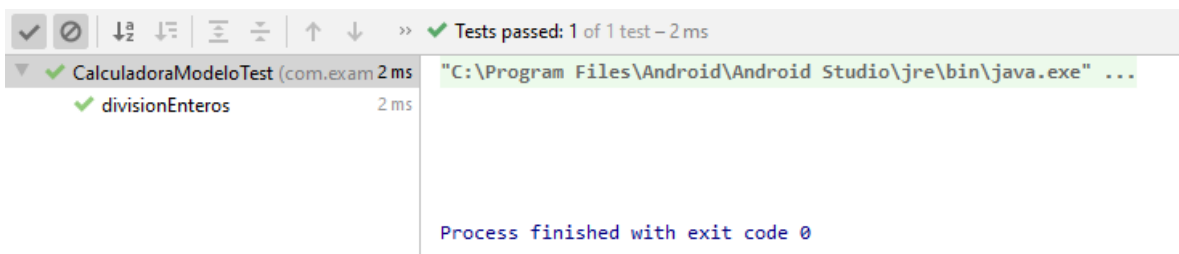
Test de resta de enteros negativos



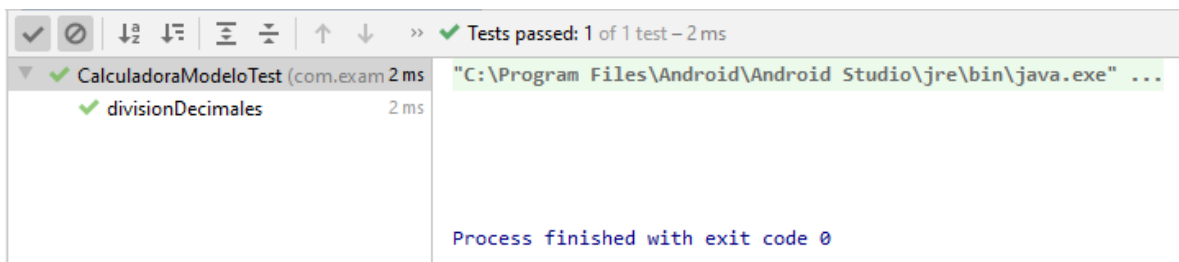
Test de resta de decimales negativos



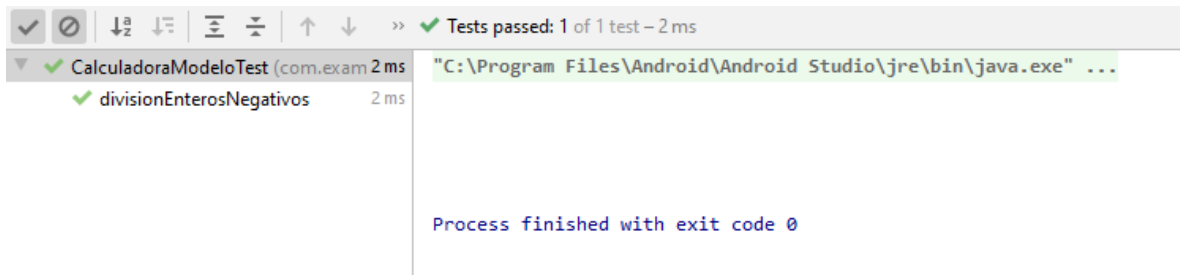
Test de division de enteros



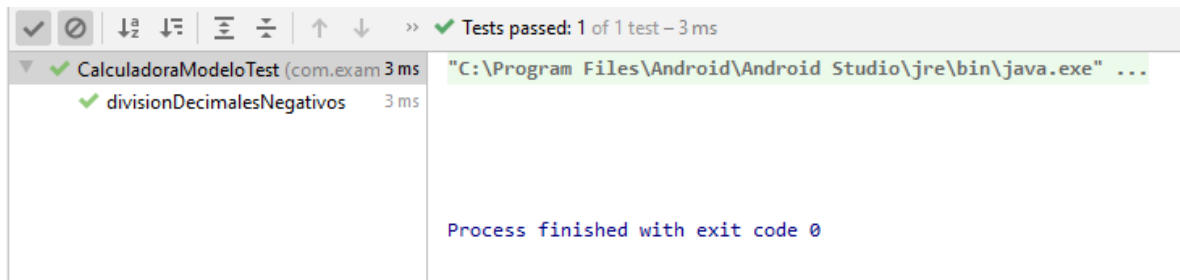
Test de division de decimales



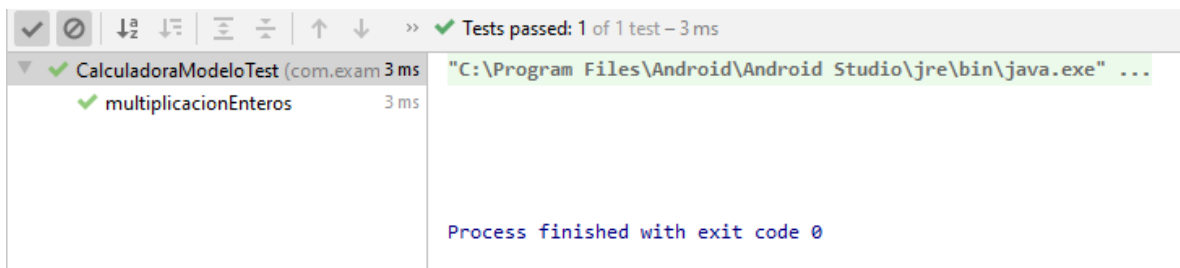
Test de division de enteros negativos



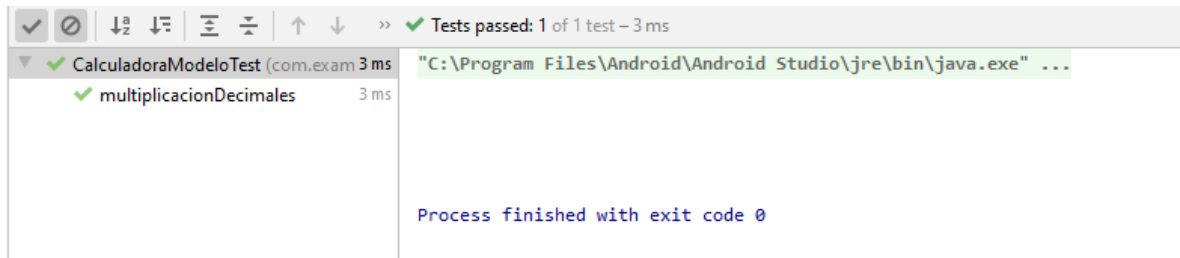
Test de division de decimales negativos



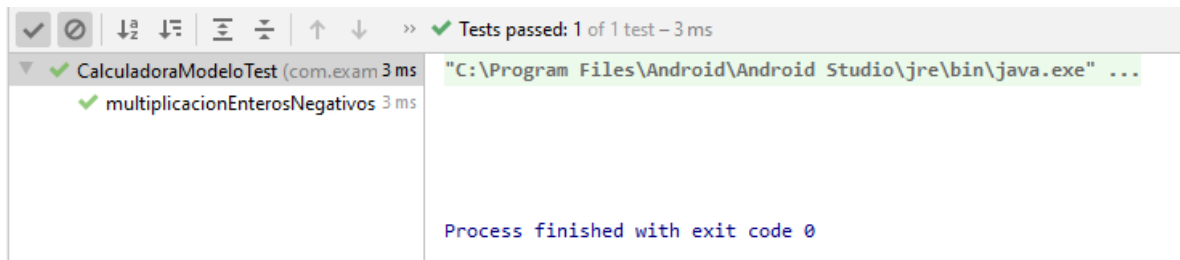
Test de multiplicación de enteros



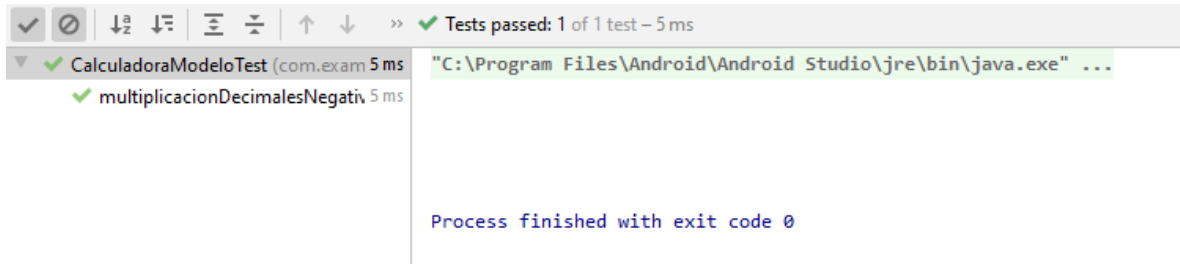
Test de multiplicación de decimales



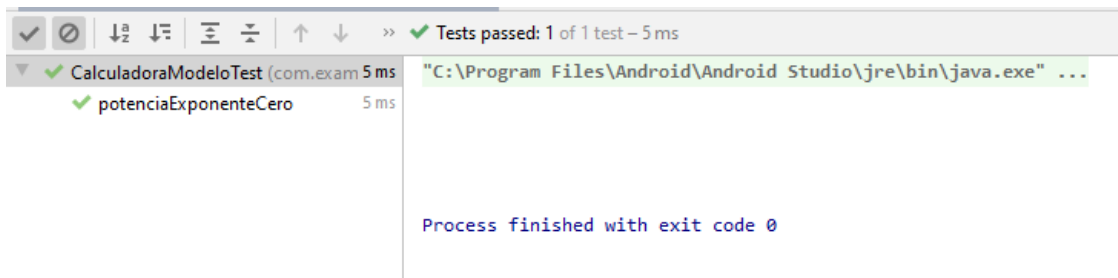
Test de multiplicación de enteros negativos



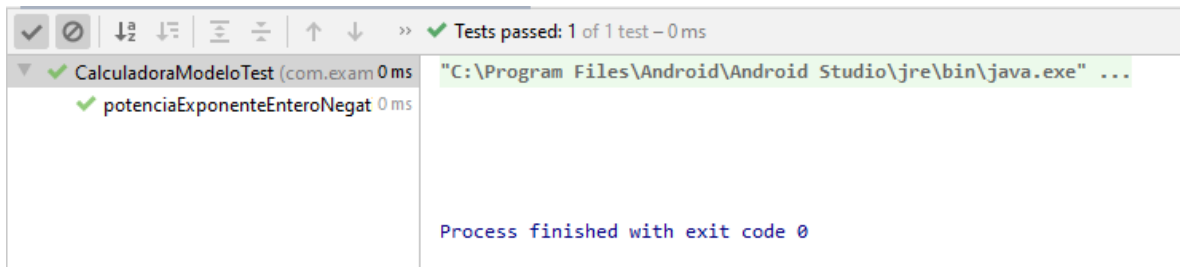
Test de multiplicación de decimales negativos



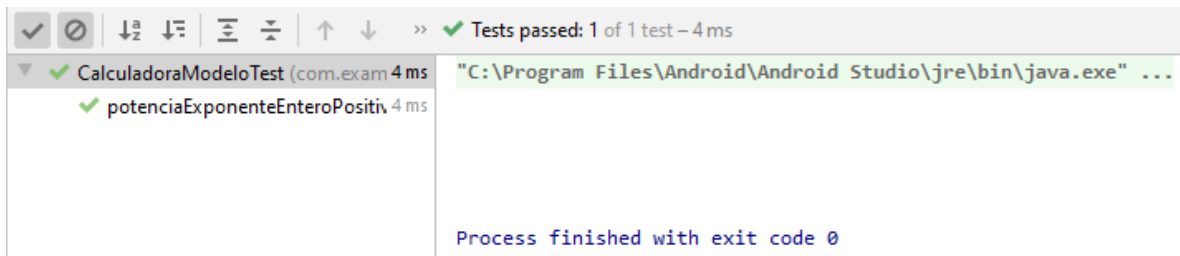
Test de potencia con exponente cero



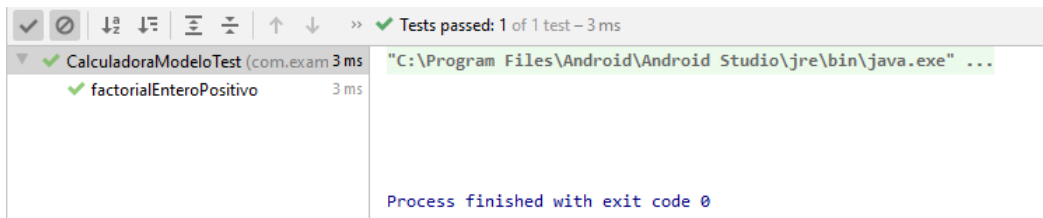
Test de potencia con exponente negativo



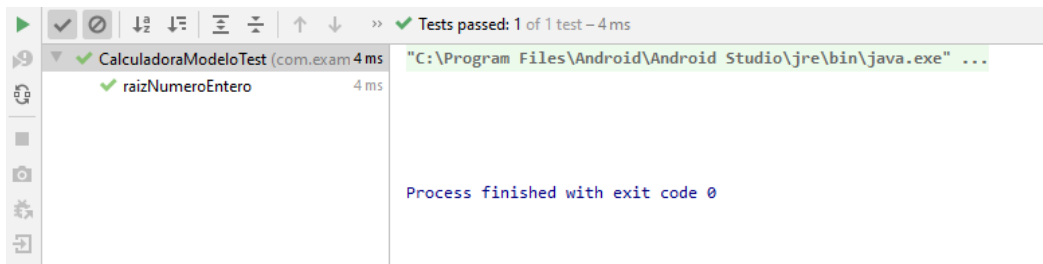
Test de potencia con exponente entero positivo



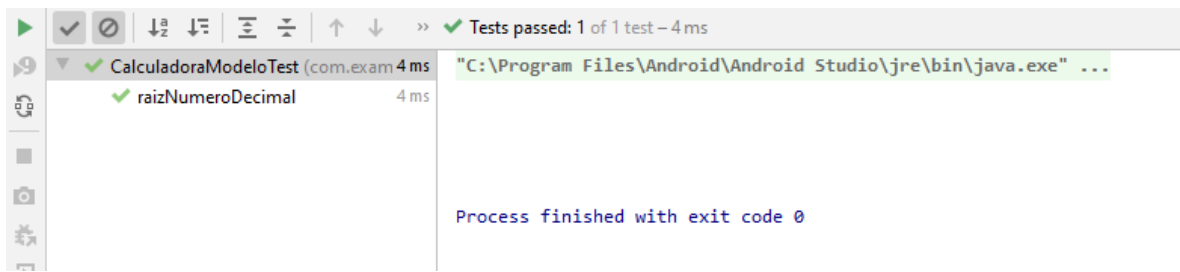
Test de factorial con entero positivo



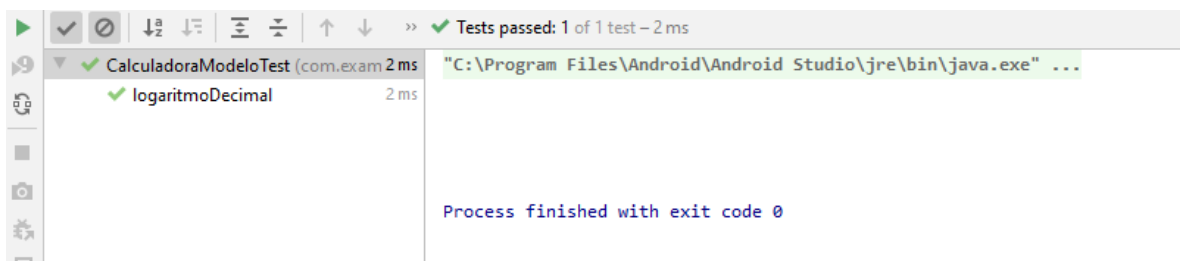
Test raiz con numero entero



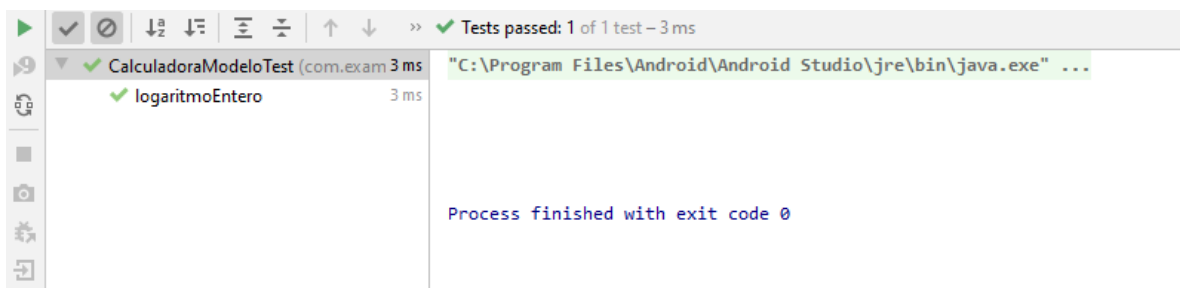
Test raíz con numero decimal



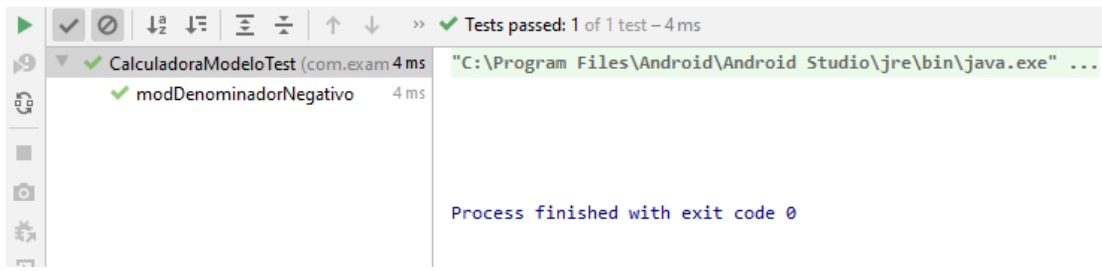
Test logaritmo decimal



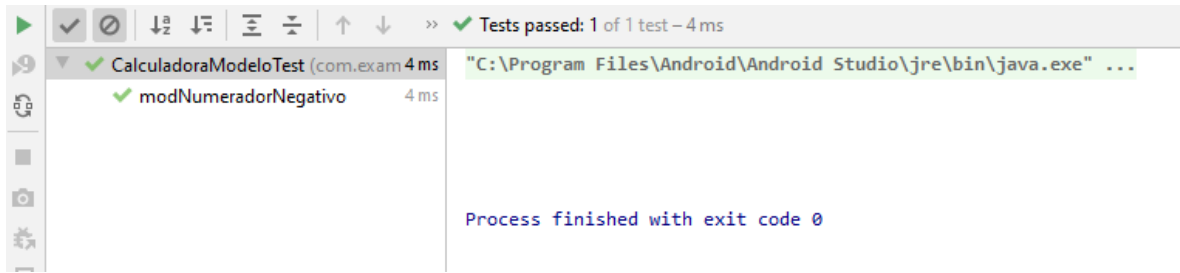
Test logaritmo entero



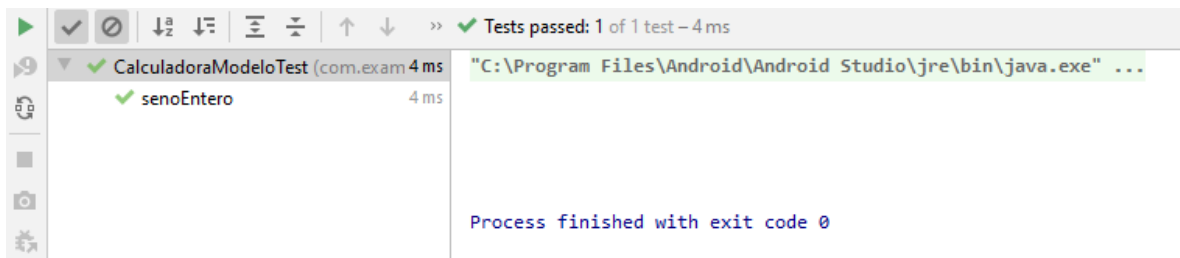
Test mod con denominador negativo



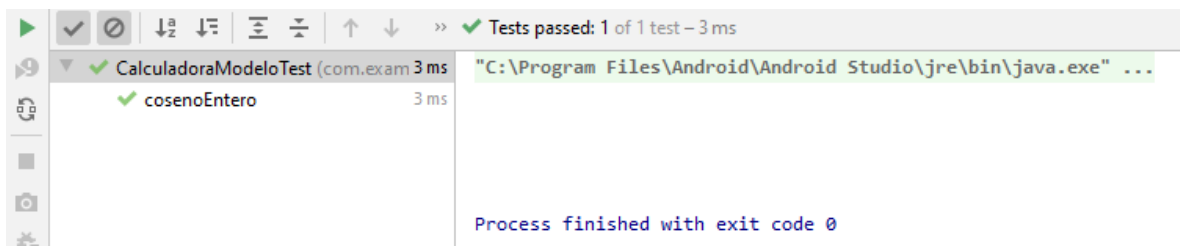
Test mod con numerador negativo



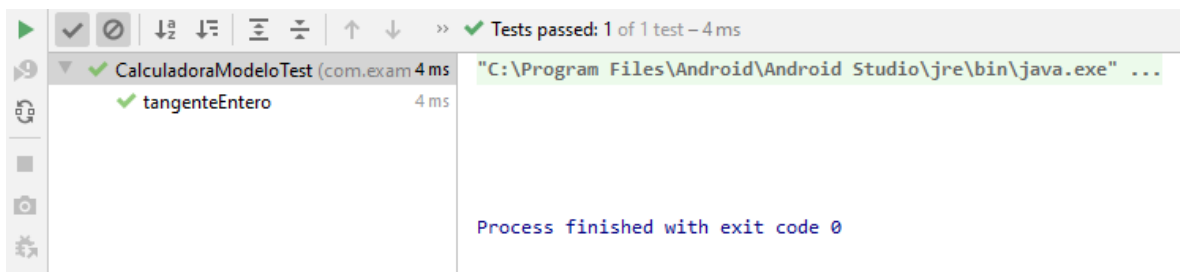
Test de seno con números enteros



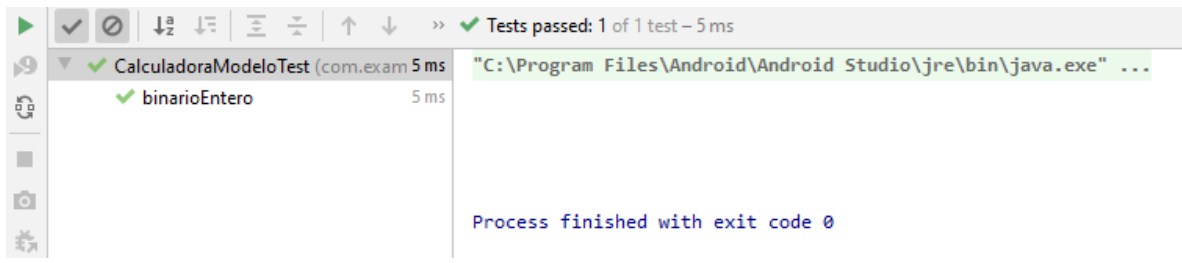
Test de coseno con números enteros



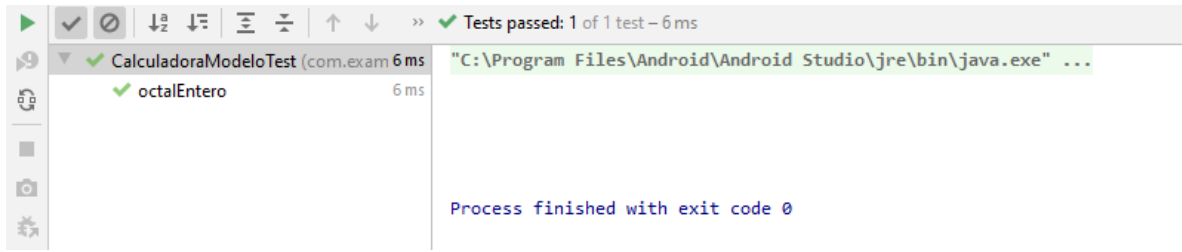
Test de tangente con números enteros



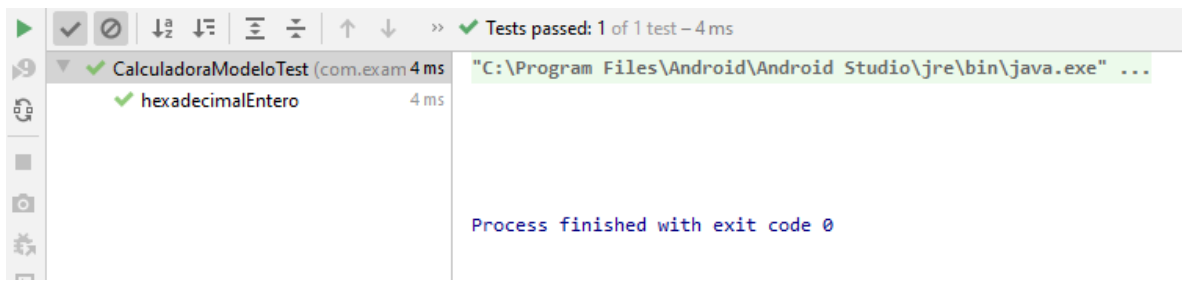
Test de binario con números decimales



Test de octal con números decimales



Test de hexadecimal con números decimales



Test de mR

