UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE
PRUEBAS DE SOFTWARE

NOMBRE: Alisson Clavijo
NRC:9870

CASOS DE PRUEBA – JUNIT5 - MOKITO

Caso de prueba	Función	Tipo de prueba	Valores de entrada	Resultado esperado	Ejecución	Observación
CP1	Dividir	AssertEquals	16 y 4	4	OK	
CP2	Factorial	AssertEquals	5!	120	NO	La función no admite el valor 5!
CP3	Potencia	AssertEquals	2^3	8	NO	La función no admite el valor 2^3
CP4	Dividir	AssertThrows	12 y 0	ArithmecticException	OK	
CP5	Factorial	AssertThows	-5!	IllegalArgumentException	NO	La función no admite el valor -5!
CP6	Potencia	AssertThows	2^1	IllegalArgumentException	NO	La función no admite el valor 2^3
CP7	Dividir	AssertEquals	774396530 y 154879306	5	OK	
CP8	Factorial	AssertEquals	12!	4790001600	NO	La función no admite el valor 12
CP9	Potencia	AssertEquals	5^-3	0.008	NO	La función no admite el valor 5^-3
CP10	Dividir	AssertThows	124 y 0.2	ArithmecticException	NO	Los vales ingresados son correctos, no deberían ocasionar un ArithmecticException ni ningún error.

UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE
PRUEBAS DE SOFTWARE

SCRIPTS - USADOS EN LOS CASOS DE PRUEBA

Caso de prueba	Función	Script usado para la función

CP1	Dividir	void dividir_ValidArguments_ReturnsCorrectResult() {
		Operaciones calculadora = new Operaciones();
		Operaciones calculadoraMock = Mockito.spy(calculadora);
		doReturn(2.0).when(calculadoraMock).dividir(10.0, 5.0);
		
		//CASO 1
		assertEquals(4, calculadoraMock.dividir(16.0, 4));
		}

CP2	Factorial	void calcularFactorial_ValidArgument_ReturnsCorrectResult() {
		Operaciones calculadora = new Operaciones();
		Operaciones calculadoraMock = Mockito.spy(calculadora);
		doReturn(6).when(calculadoraMock).calcularFactorial(3);
		
		//CASO 2
		int resultado = calculadora.calcularFactorial(5!);
		assertEquals(120, resultado);
		}

CP3	Potencia	void calcularPotencia_ValidArguments_ReturnsCorrectResult() {
		// Crear una instancia real de la clase Calculadora
		Operaciones calculadora = new Operaciones();
		
		//CASO 3
		int resultado = (int) calculadora.calcularPotencia(2^3);
		assertEquals(8, resultado);
		}

CP4	Dividir	void dividir_ValidArguments_ReturnsCorrectResult() {
		Operaciones calculadora = new Operaciones();
		Operaciones calculadoraMock = Mockito.spy(calculadora);
		doReturn(2.0).when(calculadoraMock).dividir(10.0, 5.0);
		
		//CASO 4
		assertThrows(ArithmeticException.class, () -> calculadoraMock.dividir(12, 0));
		}

CP5	Factorial	@Test
		void calcularFactorial_ValidArgument_ReturnsCorrectResult() {
		Operaciones calculadora = new Operaciones();
		Operaciones calculadoraMock = Mockito.spy(calculadora);
		doReturn(6).when(calculadoraMock).calcularFactorial(3);
		
		//CASO 5
		assertThrows(IllegalArgumentException.class, () ->
		calculadora.calcularFactorial(-5!));
		}

CP6	Potencia	void calcularPotencia_ValidArguments_ReturnsCorrectResult() {
		// Crear una instancia real de la clase Calculadora
		Operaciones calculadora = new Operaciones();
		
		//CASO 6
		assertThrows(IllegalArgumentException.class, () ->
		calculadora.calcularPotencia(2^1));
		}

CP7	Dividir	void dividir_ValidArguments_ReturnsCorrectResult() {
		Operaciones calculadora = new Operaciones();
		Operaciones calculadoraMock = Mockito.spy(calculadora);
		doReturn(2.0).when(calculadoraMock).dividir(10.0, 5.0);
		
		//CASO 7
		assertEquals(5, calculadoraMock.dividir(774396530, 154879306));
		}

CP8	Factorial	void calcularFactorial_ValidArgument_ReturnsCorrectResult() {
		Operaciones calculadora = new Operaciones();
		Operaciones calculadoraMock = Mockito.spy(calculadora);
		doReturn(6).when(calculadoraMock).calcularFactorial(3);
		
		//CASO 8
		int resultado = calculadora.calcularFactorial(12!);
		assertEquals(479001600, resultado);
		}

CP9	Potencia	void calcularPotencia_ValidArguments_ReturnsCorrectResult() {
		// Crear una instancia real de la clase Calculadora
		Operaciones calculadora = new Operaciones();
		
		//CASO 9
		int resultado = (int) calculadora.calcularPotencia(5^-3);
		assertEquals(0.008, resultado);
		}

UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE
PRUEBAS DE SOFTWARE

CP10	Dividir	void dividir_ValidArguments_ReturnsCorrectResult() {
		Operaciones calculadora = new Operaciones();
		Operaciones calculadoraMock = Mockito.spy(calculadora);
		doReturn(2.0).when(calculadoraMock).dividir(10.0, 5.0);
		
		//CASO 10
		assertThrows(ArithmeticException.class, () -> calculadoraMock.dividir(124, 0.2));
		}