**Pruebas unitarias Junit**

URL del repositorio GITHUB compartido: <https://github.com/Vasi81/Calculadora_Entornos>

Una vez creado el paquete com.entornos.junit en el proyecto Calculadora\_Entornos cada miembro del equipo realiza las pruebas con una clase creada por otro miembro del equipo.

Se hacen varias reuniones para repartir tareas, aclarar dudas y tomar decisiones.

Pruebas métodos clase Producto

Vesselin Stanev

Primer @Test

En el primer test implemento la prueba testProductoEntero que realiza un test multiplicando 2 números enteros. Para esta prueba utilizo el assertTrue que devuelve TRUE si el resultado del producto de los dos números es igual a 35

Segundo @Test

En este caso implemento la prueba testProductoReales que realiza un test multiplicando 2 números reales. Para esta prueba utilizo el **assertEquals** que comprueba si el resultado esperado (35.0) coincide con el resultado real que nos daría el método metProductoReal (), (metProductoReal(operador3, operador4). Si el resultado esperado coincide con el resultado real nos sacará por consola el siguiente mensaje:"El test del cálculo del producto de dos números reales es correcto"

}

Tercer @Test

En este caso implemento la prueba del testProductoTresReales que realiza un test multiplicando 3 números reales. Para esta prueba utilizo de nuevo el método **assertEquals** que compruebasi el resultado esperado (315) coincide con el resultado real que nos daría el método metProductoReal3(), (metProductoReal3(operador3, operador4, operador5). Si el resultado esperado coincide con el resultado real nos sacará por consola el siguiente mensaje: "El test del cálculo del producto de tres números reales es correcto"

Cuarto @Test

En este caso hago la prueba implementando testPotencia utilizando el método **fail** de Junit que hace que salte la excepción cuando el valor del operador 2 es mayor que 15. Llamamos al método metProductoPotencia dentro del bloque try-catch que captura la excepción esperada.

En caso contrario (si el valor es <=15) la operación se ejecutaría correctamente.

José María Marqués González

Primer @Test

En el primer test “TestSumaEnteros”, realiza una suma de dos números enteros utilizando “AsserEquals”. Y esperando que el resultado se igual al esperado, en este caso sería 11, sumando operador1 = 5 y operador2 = 6. Siendo los operadores mayores a cero.

Segundo @Test

En este segundo test “testSumaReal”, realiza una suma de tres números reales utilizando el método “asserEquals”. Y esperando que el resultado se igual al esperado, en este caso sería 10, sumando operador3 = 3.0, operador4 = 6.0 y operador5 = 1.0. Siendo los operadores mayores a cero.

Tercer @Test

En este tercer test “testSumaReal”, realiza una suma de dos números reales utilizando el método “asserEquals”. Y esperando que el resultado se igual al esperado, en este caso sería 8, sumando operador6 = 6.5 y operador7 = 1.5. Siendo los operadores mayores a cero.

Cuarto @Test

En este cuarto test “testUnicoentero”, realiza la suma de todos los enteros acumulados más el valor que le doy a mi variable num = 5. El resultado es el esperado 16. Los operadores mayores a cero también.

Quinto @Test

En este quinto test “testUnicoReal”, realizo la suma de todos los números acumulados reales, y les sumo al valor num = 1.0 que le doy a mi variable. El resulta do es 11 que es el esperado. Los operadores mayor a cero.