Unión Europea
Fondo Social Europeo
El FSE invierte en tu futuro

Nombre Apellido:	Lei Wang	NOTA	
Módulo:	Entornos de desarrollo	Fecha:	11/05/2025
C.F:	DAM_SM	Curso:	1º
Profesorado:	Joan Agustí Suàrez		
UD3 y UD4:	Pruebas, refactorización y JavaDoc		11:15-13:10

NOTA: En el bloque examen tercera evaluación (esta vez no es terrible) tenéis disponible el fichero .java del que se habla a continuación, además la tarea donde entregar los documentos que generéis.

```java
public class Calculadora {

    public int operar(String operador, int a, int b) {
        if (operador.equals("sumar")) {
            System.out.println("🔧 Iniciando operación: SUMAR");
            int resultado = a + b;
            System.out.println("Resultado: " + resultado);
            return resultado;
        } else if (operador.equals("restar")) {
            System.out.println("🔧 Iniciando operación: RESTAR");
            int resultado = a - b;
            System.out.println("Resultado: " + resultado);
            return resultado;
        } else if (operador.equals("multiplicar")) {
            System.out.println("🔧 Iniciando operación: MULTIPLICAR");
            int resultado = a * b;
            System.out.println("Resultado: " + resultado);
            return resultado;
        } else if (operador.equals("dividir")) {
            System.out.println("🔧 Iniciando operación: DIVIDIR");
            if (b == 0) {
                System.out.println("⚠️ Error: División por cero");
                throw new ArithmeticException("División por cero");
            }
            int resultado = a / b;
            System.out.println("Resultado: " + resultado);
            return resultado;
        } else {
            System.out.println("⚠️ Error: Operación no válida");
            return 0;
        }
    }
}
```

Actividad 1. Caja negra. (La cantidad de filas es orientativa, no quiere decir que tengáis que hacer 7)

Prueba	Resultado esperado	Resultado obtenido	Conclusión
Sumar 2+3	Resultado: 5	Resultado: 5 Resultado: 5	Se ha imprimido 2 veces el resultado
Restar 6-3	Resultado: 3	Resultado: 3 Resultado: 3	Se ha imprimido 2 veces el resultado
Restar 3-6	Resultado: -3	Resultado: -3 resultado: -3	Se ha imprimido 2 veces el resultado
Multiplicar 6x3	Resultado: 18	Resultado: 18 resultado: 18	Se ha imprimido 2 veces el resultado
Dividir 6/3	Resultado: 2	Resultado: 2 Resultado: 2	Se ha imprimido 2 veces el resultado
Cualquier operación con mayus al principio (ejemplo: Sumar con la s en mayus)	Resultado: x	⚠ Error: Operación no válida Resultado: 0	No funciona al poner cualquier letra de la operación en mayuscula

Actividad 2. Caja Blanca. Dibuja el grafo (diagrama de nodos como algunos le llamáis) marcando claramente los nodos predicado y calcula la complejidad ciclomática.

Rellena una tabla con los diferentes caminos que pueden existir.

Sale 2 veces el resultado porque hay un trozo de codigo que es un try y un catch y eso nos muestra resultado, y luego en el if-else sale otra vez el resultado

el try catch es para por si pones un carácter incorrecto y te lance la excepcion

nodo predicado 1 4 posibles caminos

nodo predicado 2 2 posibles caminos

nodo predicado 3 5 posibles caminos

nodo predicado 4 2 posibles caminos

complejidad ciclomatica: 56 posibles caminos en total contando excepciones

diagrama con archivo a parte

Actividad 3. JUnit. Realiza las pruebas unitarias necesarias para cubrir el 100% de este código.

Actividad 4. Aplica todos los patrones de **refactorización** que detectes. Enumera los patrones que has utilizado y el porqué.

Voy a quitar el try-catch (hace que imprima 1 vez el resultado pero lo voy a poner de otra forma)

en vez de tantos if, if-else, else voy a poner un switch, con un metodo para cada operación

ponemos a y b como atributos y los iniciamos

hacemos metodos para cada operación (en el metodo de division tambien ponemos la comprobacion de si b == 0)

luego ya en el main ponemos un switch con cada operación para que haga la operación requerida y luego imprimimos

hay una clase Calculadora que es la inicial

y luego el refactorizado se llama CalculadoraRefactorizado

Actividad 5. Documenta el código que has obtenido lo máximo que puedas e indica a continuación los pasos a seguir si queremos generar el JavaDoc.

La estructura de javadoc es asi
/**

*

*

*

*/

/** es para iniciar el javadoc

cada * es una linea de comentario

*/ es para cerrar el javadoc