PRÁCTICA 8. jUnit | ***2 H (1 Ses)***

Objetivos:

- Aprender a realizar una bateria de test
- Repasar y profundizar en la realización de jUnit

Contenido:

Ejercicio 1: Bateria de test de Calculator (**25m**)

Dado el siguiente código (se encuentra en bitbucket) se pide realizar la batería de test (se realizó en la sesión 1).

```
public class Calculator {
	public double add(double a, double b) {
	return a + b;
	}
	public double subtract(double a, double b) {
		return a - b;
	}
	public double multiply(double a, double b) {
		return a * b;
	}
	public double divide(double a, double b) {
		if (b == 0) {
			throw new ArithmeticException();
		}
		return a / b;
	}
}
```

Antes de realizar los ejercicios se deben configurar el proyecto para el uso de jUnit. Pasos:

1. Creamos un nuevo paquete o un nuevo proyecto
2. Añadimos el código anterior Calculator.java
3. El código fuente de la bateria de test se encuentran en una carpeta distinta del código fuente. Para ello nos crearemos una nueva "carpeta de código fuente" (source folder). **Importante**: se crea un source folder (que se incorpora en el proceso de compilación del proyecto) y no un folder (que no se integra)
    1. File → New Source Folder →
    2. Lo denominamos "test"
4. Para el proceso de creación de la bateria se puede realizar manualmente o eclipse se puede generar el esqueleto de la bateria. Para ello
    1. A nivel de proyecto → File → New → jUnit Test Case
    2. En "source folder" → test
    3. En "name" → TestCalculator
    4. En "Class under test" → Calculator. Pulsamos next y seleccionamos todos los métodos que se desean realizar la bateria de test.

Ejercicio 2: Bateria test de Persona (**30m**)

Se proporciona en bitbucket el código fuente de una clase denominada Persona.java. Se pide generar el esqueleto de su batería de test e implementar la batería de todos los métodos. Para el método calcularEdad() se debe realizar los siguientes casos:

- Una fecha que devuelva como resultado 1 año
- Una fecha que devuelva como resultado 1 año, siendo la fecha el día antes de su cumpleaños.
- Una fecha que devuelva como resultado 2 años, siendo la fecha el día de su cumpleaños.
- Una fecha que devuelva como resultado 2 años, siendo la fecha varios meses después de su cumpleaños.
-

Ejercicio 3: Bateria test de EjercicioArray (**30m**)

Se proporciona en bitbucket el código fuente de una clase denominada EjercicioArray.java con varios algoritmos sobre un array de enteros. Se pide generar el esqueleto de su batería de test e implementar la batería de todos los métodos. La forma de llamar a cada uno de los métodos es: EjercicioArray.no23(new int[]{5,6,8,10}); donde se le está proporcionando los valores del array estáticamente.

Ejercicio 4: Bateria test de Desguace (**30m**)

A partir de la sesión 7 de Desguace (con excepciones) se pide realizar la batería de test de Desguace. Para ello se deberá seguir los siguientes pasos:

1. Para el proceso de creación de la bateria se puede realizar manualmente o eclipse se puede generar el esqueleto de la bateria. Para ello
    1. A nivel de proyecto → File → New → jUnit Test Case
    2. En “source folder” → test
    3. En “name” → TestDesguace
    4. En “Class under test” → Calculator. Pulsamos next y seleccionamos todos los métodos que se desean realizar la bateria de test.

Habrá que tener en cuenta:

- testDesguace(): ¿Cómo se realiza una bateria de un constructor?
- testGetNombre(): Batería de getter del nombre: Recuperar y ver si coincide con lo establecido.
- testSetNombre(): : Batería de setter del nombre. Modificar el valor inicial y ver si coincide.
- testGetVehiculoBastidor(): ¿Qué comprobaciones se debe de hacer a la hora de buscar un vehículo por el bastidor? Que no exista el bastidor y que exista. ¿Qué sucede con la excepción cuando es devuelta?
- TestAddVehiculo(): Debe realizarse la inserción de un vehículo que exista y de otro que no exista.
- TestAddPiezaVehiculo(): La batería más compleja. Pensad en las distintas opciones que puede haber.

- TestGetPiezaVehiculo() . Debemos controlar todos los errores posibles: ¿Qué haya bastidor? ¿Qué haya pieza?….
- TestGetPiezaDesguace(): Comprueba si un id de pieza esta en el desguace.
- TestAddPiezaDesguace(). Piensa….
- TestAddProveedor(): Piensa….
- TestGetProveedor(): Piensa….
- TestAddEmpleado(): Piensa….
- TestGetEmpleado() : Piensa….