PRÁCTICA SESIÓN 16: Introducción a las pruebas, verificación vs. validación

Ejercicio 1: Verificación (prueba unitaria)

Comprobar que un método hace *exactamente* lo que promete.

```
public class Calculadora {

   public static int dividir(int a, int b) {
      return a / b;
   }
}
```

Preguntas de verificación:

- ¿Qué pasa si b vale 0?
- ¿El método cumple su contrato para *todos* los enteros?
- ¿Debería lanzar una excepción controlada?

Ejercicio 2: Verificación con corrección

Mejorar el diseño técnico.

Modifica el método para que:

- No permita dividir entre 0
- Informe claramente del error

```
public static int dividir(int a, int b) {
   if (b == 0) {
      throw new IllegalArgumentException("No se puede dividir entre
cero");
   }
   return a / b;
}
```

Ejercicio 3: Integración (encajan las piezas)

```
public class ServicioCalculo {
```

```
    public static String resultadoDivision(int a, int b) {
        int resultado = Calculadora.dividir(a, b);
        return "Resultado: " + resultado;
    }
}
```

Prueba de integración:

- ¿Qué ocurre si dividir() lanza excepción?
- ¿Este método la gestiona o la propaga?
- ¿El mensaje siempre es coherente?

Aquí no pruebas una pieza, pruebas cómo colaboran.

Ejercicio 4: Validación

Imagina que esto se muestra en una app educativa.

Preguntas de validación:

- ¿Un usuario entiende “IllegalArgumentException”?
- ¿Es mejor mostrar “No se puede dividir entre cero”?
- ¿El formato del resultado es claro?

Ejercicio 5 – Prueba de sistema (flujo completo)

```
public class Main {
    public static void main(String[] args) {
        System.out.println(ServicioCalculo.resultadoDivision(10, 2));
    }
}
```

Cambia valores y observa:

- (10, 2)
- (10, 0)
- (-10, 2)

Preguntas:

- ¿El programa se rompe?
- ¿El fallo llega al usuario?
- ¿El comportamiento es aceptable?