

Ejercicios Tema 5. Diseño y realización de pruebas

Bloque 1. Rompe el programa

Observa el siguiente método:

```
public static int calcularPrecioFinal(int precioBase, int descuento) {  
    return precioBase - (precioBase * descuento / 100);  
}
```

Ejercicio 1

Diseña al menos 7 casos de prueba distintos para el método `calcularPrecioFinal`

Para cada uno indica:

- Entrada
- Resultado esperado
- Oráculo

1

Entrada: `precioBase = 100, descuento = 20`

Resultado esperado: 80

Oráculo: $100 - (100 \times 20 / 100) = 100 - 20 = 80$

2

Entrada: `precioBase = 200, descuento = 0`

Resultado esperado: 200

Oráculo: $200 - (200 \times 0 / 100) = 200$

3

Entrada: `precioBase = 150, descuento = 100`

Resultado esperado: 0

Oráculo: $150 - (150 \times 100 / 100) = 150 - 150 = 0$

4

Entrada: `precioBase = 100, descuento = 150`

Resultado esperado: -50

Oráculo: $100 - (100 \times 150 / 100) = 100 - 150 = -50$

5

Entrada: `precioBase = 100, descuento = -10`

Resultado esperado: 110

Oráculo: $100 - (100 \times (-10) / 100) = 100 - (-10) = 110$

6

Entrada: `precioBase = 0, descuento = 50`

Resultado esperado: 0

Oráculo: $0 - (0 \times 50 / 100) = 0$

7

Entrada: `precioBase = -100, descuento = 10`

Resultado esperado: -90

Oráculo: $-100 - (-100 \times 10 / 100) = -100 - (-10) = -90$

Ejercicio 2

Contesta las siguientes preguntas relacionadas con el método calcularPrecioFinal

¿Qué ocurre si descuento es negativo? **El precio final aumenta en lugar de disminuir.**

¿Y si es mayor que 100? **El resultado puede ser negativo**

¿Crees que el método debería permitir valores negativos o mayores que 100? **No, los descuentos no pueden ser de más del 100%**

Ejercicio 3

Propón una mejora del método calcularPrecioFinal que haga que su comportamiento sea más seguro.

Bloque 2. El cazador de bugs

Observa el siguiente método:

```
public static int maximo(int[] datos) {  
    int max = 0;  
    for (int i = 0; i < datos.length; i++) {  
        if (datos[i] > max) {  
            max = datos[i];  
        }  
    }  
    return max;  
}
```

Ejercicio 4

Encuentra al menos 3 entradas de datos donde el método maximo falle.

Para cada entrada indica:

- Resultado obtenido
- Resultado correcto
- Tipo de fallo

`int[] datos = {-5, -10, -3};` (VARIOS NEGATIVOS)

Resultado obtenido: 0

Resultado correcto: -3

Tipo de fallo: Como se inicializa a 0 nunca van a salir como máximos

`int[] datos = {-7};` (UN NEGATIVO)

Resultado obtenido: 0

Resultado correcto: -7

Tipo de fallo: Como se inicializa a 0 nunca van a salir como máximos

`int[] datos = {};` (VACÍO)

Resultado obtenido: 0

Resultado correcto: Debería dar error o avisar

Tipo de fallo: No avisa porque no está contemplado

Ejercicio 5

Escribe una versión correcta del método maximo

Bloque 3. Diseña el oráculo

Observa el siguiente método:

```
public static int[] ordenar(int[] datos) {  
    // algoritmo desconocido  
}
```

Ejercicio 6

Define 3 propiedades que siempre debe cumplir el resultado del método ordenar.

Tiene que estar ordenado

Tiene que ser igual de largo, no puede añadir ni quitar números.

Y tienen que ser los mismos números (solo tiene que moverlos)

Ejercicio 7

Explica por qué en el método ordenar es mejor usar oráculos por propiedades que valores concretos.

Porque permiten comprobar que el resultado cumple las condiciones necesarias sin depender de valores concretos. Esto es mejor cuando se trabaja con arrays grandes, ya que calcular manualmente el resultado sería lento y podría dar errores.

Bloque 4. Caminos y decisiones

A la vista del siguiente método:

```
public static String clasificarEdad(int edad) {  
    if (edad < 0) {  
        return "ERROR";  
    } else if (edad < 12) {  
        return "NIÑO";  
    } else if (edad < 18) {  
        return "ADOLESCENTE";  
    } else {  
        return "ADULTO";  
    }  
}
```

Ejercicio 8

Calcula el número mínimo de tests necesarios para el método clasificarEdad. Después propón tantos tests como hayas obtenido.

Como tiene 4 posibles resultados, los tests minimos seran 4, uno para cada resultado

Test 1: edad = -3 → "ERROR"

Test 2: edad = 5 → "NIÑO"

Test 3: edad = 15 → "ADOLESCENTE"

Test 4: edad = 25 → "ADULTO"

Bloque 6. Prioriza como un profesional

Se quiere desarrollar una aplicación Web de compra online.

Ejercicio 10

Ordena de mayor a menor prioridad:

- Login
- Mostrar catálogo
- Pago con tarjeta
- Cambiar avatar
- Recuperar contraseña

1. Login
2. Mostrar catálogo
3. Pago con tarjeta
4. Recuperar contraseña
5. Cambiar avatar