

Guía de ejercicios 3:

Temas abarcados:

- Genericidad
- Excepciones
- Algoritmos de búsqueda y ordenamiento
- Recursividad

Ejercicio 1:

Implementá una clase genérica `Caja<T>` que permita almacenar un objeto, obtenerlo (`getContenido`) y reemplazarlo (`setContenido`). Probala con `String`, `Integer` y una clase propia `Libro`.

Ejercicio 2:

Escribí un método genérico `public static <T> void imprimirArray(T[] arr)` que imprima los elementos del arreglo. Probalo con un arreglo de enteros, uno de cadenas y otro de objetos `Persona`

Ejercicio 3:

Creá una clase genérica `Contador<T>` que cuente cuántas veces aparece un valor específico dentro de un arreglo. Si el arreglo es `null`, lanzá una excepción personalizada `ArregloNuloException`.

Ejercicio 4:

Pedile al usuario que ingrese dos números y mostrá el resultado de la división. Si se intenta dividir por cero, capturá la excepción y mostrá un mensaje personalizado.

Ejercicio 5:

Escribí un programa que lea una lista de nombres desde un arreglo y verifique que tengan más de 3 letras. Si alguno no cumple, arrojá una excepción y registrala en un archivo de log (`log.txt`).

Ejercicio 6:

Implementá y compará los algoritmos de **selección**, **inserción** y **shell sort** ordenando una lista de temperaturas. Mostrá la cantidad de iteraciones realizadas por cada método.

Ejercicio 7:

Implementá los algoritmos de **búsqueda secuencial** y **binaria** para buscar un número ingresado por el usuario en un arreglo de enteros. Indicá si fue encontrado o no y en qué posición.

Ejercicio 8:

Escribí un método recursivo que reciba un número entero y devuelva la suma de sus dígitos. Ej: `sumaDigitos(1234)` → 10.

Ejercicio 9:

Escribí un método recursivo que cuente la cantidad de vocales en una cadena. Ej: `contarVocales("Recursividad")` → 5.

Ejercicio 10:

Implementá el clásico problema de las **Torres de Hanoi** para n discos. Mostrá cada movimiento necesario para resolverlo.