# Práctico 8: Genéricos

## Objetivo:

**Desarrollar habilidades en el uso de Genéricos en Java para mejorar la seguridad, reutilización y escalabilidad del código. Comprender la implementación de clases, métodos e interfaces genéricas en estructuras de datos dinámicas. Aplicar comodines (?, extends, super) para gestionar diferentes tipos de datos en colecciones. Utilizar Comparable y Comparator para ordenar y buscar elementos en colecciones de manera flexible. Integrar Genéricos en el diseño modular del software, optimizando su organización y mantenimiento.**

***Resultados de aprendizaje:***

1. **Aplicar Generics en Java**: El estudiante será capaz de utilizar tipos genéricos en clases, métodos e interfaces, comprendiendo su importancia en la seguridad y reutilización del código dentro de colecciones y estructuras de datos.
2. **Gestionar y manipular colecciones genéricas**: El estudiante será capaz de implementar clases y métodos genéricos como `Sorteador` y `Buscador`, aplicar comodines (`?`, `extends`, `super`) y diseñar estructuras eficientes para manejar datos dinámicos.
3. **Ordenar y buscar elementos en colecciones genéricas**: El estudiante será capaz de emplear `Comparable` y `Comparator` para definir criterios de ordenación personalizados en colecciones, además de desarrollar mecanismos genéricos de búsqueda en listas.
4. **Integrar Generics en el diseño modular del software**: El estudiante será capaz de estructurar código reutilizable y escalable mediante el uso de Generics en interfaces y métodos, optimizando la organización y flexibilidad del desarrollo en Java.

## ¿Qué es una Kata y cómo se utiliza en programación?

**Una kata es un ejercicio de programación diseñado para mejorar habilidades de codificación mediante la repetición y el aprendizaje progresivo.** El término proviene de las artes marciales, donde las katas son secuencias de movimientos que se practican repetidamente para perfeccionar la técnica.

En programación, las katas ayudan a los programadores a reforzar conceptos, mejorar la comprensión del código y desarrollar buenas prácticas. **Se recomienda resolver una kata varias veces, intentando mejorar el código en cada iteración, utilizando mejores estructuras, nombres más claros y principios de diseño.**

## Importante:

**Intentar resolver cada kata sin mirar la solución.**

**Comprobar la solución y corregir errores si es necesario.**

**Repetir 2 o 3 veces para mejorar la comprensión, lógica y el código.**

**Experimentar con diferentes valores para reforzar el aprendizaje.**

## Resolver Katas

**A continuación, te presento 4 enunciados de katas en Java para fortalecer los conceptos vistos en el módulo 8.**

**Kata 1**: Introducción a **clases genéricas**.

**Kata 2**: Uso de **genéricos en colecciones**.

**Kata 3**: Aplicación de **Comparable y Comparator** en objetos genéricos.

**Kata 4**: Uso avanzado de **interfaces y métodos genéricos** en búsquedas.

---

### Kata 1: Manejo de Productos en un Pedido (Nivel Básico)

**Enunciado:**

Crea una clase genérica `Producto<T>` que permita manejar productos con diferentes tipos de identificadores (por ejemplo, `String` para códigos SKU o `Integer` para IDs internos). Debe incluir métodos para obtener el identificador y el precio.

**Clases y/o interfaces**

**Clase:** `Producto<T>`

**Atributos**

- `private T id;`
- `private String nombre;`
- `private double precio;`

**Métodos**

- `public Producto(T id, String nombre, double precio);`
- `public T getId();`
- `public String getNombre();`
- `public double getPrecio();`
- `public void setPrecio(double precio);`
- `public String toString();`

🎯 **Tarea a realizar**

- Implementar la clase `Producto<T>`.
- Crear una lista de productos con diferentes tipos de identificadores.
- Imprimir la lista de productos.

---

## Kata 2: Carrito de Compras Genérico (Nivel Intermedio)

**Enunciado:**
Crea una clase `Carrito<T>` que almacene productos genéricos y permita agregar, eliminar y obtener el total del carrito.

**Clases y/o interfaces**

**Clase:** `Carrito<T extends Producto<?>>`

**Atributos**

- `private List<T> productos;`

**Métodos**

- `public void agregarProducto(T producto);`
- `public void eliminarProducto(T producto);`
- `public double obtenerTotal();`
- `public void mostrarProductos();`

🎯 **Tarea a realizar**

- Implementar la clase `Carrito<T>`.

- Agregar productos al carrito y calcular el total.
- Mostrar el contenido del carrito.

---

## Kata 3: Comparación y Ordenación de Pedidos (Nivel Avanzado)

**Enunciado:**

Implementa la interfaz `Comparable<T>` en la clase `Pedido` para ordenar pedidos según el total. También, utiliza un `Comparator<Pedido>` para ordenarlos por fecha.

**Clases y/o interfaces**

**Clase:** `Pedido implements Comparable<Pedido>`

**Atributos**

- `private int id;`
- `private List<Producto<?>> productos;`
- `private LocalDate fecha;`

**Métodos**

- `public Pedido(int id, LocalDate fecha);`
- `public void agregarProducto(Producto<?> producto);`
- `public double calcularTotal();`
- `public int compareTo(Pedido otro);`
- `public String toString();`

**Interfaz Comparator**

- `public class ComparadorPedidosPorFecha implements Comparator<Pedido>`

🎯 **Tarea a realizar**

- Implementar `Comparable<Pedido>` para ordenar por total.
- Implementar `Comparator<Pedido>` para ordenar por fecha.
- Crear una lista de pedidos y ordenarla por total y por fecha.

## Kata 4: Búsqueda Genérica de Pedidos (Nivel Experto)

**Enunciado:**

Crea una clase `Buscador<T, K>` que permita buscar un pedido en una lista por su identificador.

**Clases y/o interfaces**

**Clase:** `Buscador<T, K>`

**Métodos**

- `public T buscar(Collection<? extends T> elementos, K id);`

**Interfaz:** `Identificable<K>`

**Métodos**

- `K getID();`
- `boolean tieneMismoID(K id);`

**Clase:** `Pedido implements Identificable<Integer>`

🎯 **Tarea a realizar**

- Implementar la clase `Buscador<T, K>`.
- Buscar un pedido en una lista de pedidos por su ID.
- Probar la funcionalidad con diferentes tipos de productos y pedidos.

---

Con estas katas, los alumnos reforzarán el uso de **genéricos en Java**, incluyendo **clases, listas, comparaciones y búsquedas genéricas**, siguiendo un enfoque progresivo. 🚀