Práctico 4: Orientación a Objetos

Objetivo:

Comprender y aplicar conceptos de Programación Orientada a Objetos en Java, incluyendo el uso de this, constructores, sobrecarga de métodos, encapsulamiento y atributos y métodos estáticos, para mejorar la modularidad, reutilización y diseño del código.

Resultados de aprendizaje:

1. **Referencia y gestión del objeto actual:** Utilizar this para referenciar la instancia actual, desambiguar atributos y mejorar la claridad del código.
2. **Inicialización y sobrecarga de constructores:** Definir y aplicar constructores para inicializar objetos, implementando sobrecarga para permitir múltiples formas de instanciación.
3. **Definición y reutilización de métodos:** Aplicar la sobrecarga de métodos para definir múltiples versiones con distintos parámetros, mejorando la flexibilidad y modularidad del código.
4. **Representación y depuración de objetos:** Implementar el método `toString` para generar representaciones legibles de objetos, facilitando su visualización y depuración.
5. **Atributos y métodos estáticos:** Diferenciar entre variables de instancia y de clase, aplicando `static` para compartir valores y funciones entre objetos.

¿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 el Caso: Gestión de Empleados

Debes modelar una clase Empleado que represente a un empleado en una empresa. La clase deberá incluir atributos esenciales, constructores sobrecargados, métodos sobrecargados, y el uso de miembros estáticos para llevar un conteo de empleados creados.

Atributos:

- id (entero): Identificador único del empleado.
- nombre (cadena): Nombre completo del empleado.
- puesto (cadena): Puesto o cargo que desempeña el empleado.
- salario (double): Salario actual del empleado.
- totalEmpleados (static entero): Contador que registra la cantidad de empleados creados.

📌 Requerimientos:

1. Uso de this:

- En los constructores, utiliza this para diferenciar entre los parámetros recibidos y los atributos de la clase.

2. Inicialización y Sobrecarga de Constructores:

- Implementa al menos dos constructores:

  * Uno que reciba todos los atributos como parámetros.

*** Otro que reciba únicamente nombre y puesto, asignando un id automático y un salario por defecto.**

- En cada constructor, incrementa el contador estático totalEmpleados.

3. Definición y Reutilización de Métodos (Sobrecarga):

- Crea un método actualizarSalario sobrecargado:

*** Una versión que reciba un porcentaje de aumento.**

*** Otra versión que reciba una cantidad fija a aumentar.**

4. Representación y Depuración de Objetos:

- Sobrescribe el método toString() para retornar una cadena con la información completa del empleado (id, nombre, puesto y salario).

5. Atributos y Métodos Estáticos:

- Implementa un método mostrarTotalEmpleados() (estático) que retorne el total de empleados creados.

Tareas a Realizar:

1. Implementa la clase Empleado en Java aplicando los conceptos mencionados.

2. Crea una clase de prueba (con método main) que:

- Instancie varios objetos Empleado utilizando ambos constructores.
- Utilice los métodos sobrecargados para actualizar los salarios de algunos empleados.
- Imprima en consola la información de cada empleado mediante toString().
- Muestre el total de empleados creados utilizando el método estático mostrarTotalEmpleados().

Consejos:

- Reflexiona sobre la importancia de utilizar this para evitar confusiones entre parámetros y atributos.

- Experimenta creando distintos escenarios con los constructores y métodos sobrecargados.

- Verifica la legibilidad de la salida del método toString() y que el contador estático se actualice correctamente.
