🚧 RETO 2. Optimización algorítmica del procesamiento de eventos de seguridad

Tipo de reto: Formativo – Proyecto abierto (enfoque algorítmico)
Unidad: Árboles balanceados y estructuras eficientes en ciberseguridad

1. Descripción del reto

En este reto, el estudiante deberá rediseñar el sistema desarrollado en el Reto 1, incorporando **estructuras de datos auto-balanceadas** que permitan mantener eficiencia en escenarios de alta carga.

A diferencia del reto anterior, el enfoque no es solo comparar algoritmos, sino **garantizar eficiencia mediante diseño estructural**, implementando explícitamente mecanismos de balanceo.

El estudiante deberá demostrar que comprende cómo la estructura de datos impacta directamente en el rendimiento de un sistema de análisis de seguridad.

2. Problema a resolver

Un sistema SIEM recibe eventos en tiempo real y necesita:

- insertar eventos dinámicamente
- mantenerlos ordenados
- permitir búsquedas rápidas

El problema crítico es que:

- los datos pueden llegar **ordenados o semi-ordenados**
- un BST puede degradarse a **O(n)**
- esto genera cuellos de botella en sistemas reales

El estudiante deberá diseñar una solución que **garantice eficiencia en todos los casos**, evitando degradaciones.

3. Objetivos de aprendizaje

El estudiante será capaz de:

- implementar un **árbol AVL desde cero**
- aplicar rotaciones (simples y dobles)
- analizar el impacto del balanceo en complejidad
- comparar estructuras bajo condiciones reales
- justificar decisiones de diseño algorítmico

4. Producto esperado

Un **proyecto en Google Colab** que incluya:

- implementación completa de:
    - BST
    - AVL (con rotaciones)
- simulación de carga dinámica
- comparación experimental
- análisis técnico profundo

5. Datos de entrada

Se deben utilizar datos similares al Reto 1, pero con énfasis en:

- inserciones en orden creciente (para forzar desbalance)
- inserciones aleatorias
- volumen escalable (hasta 1 millón)

Ejemplo de clave de indexación:

- timestamp
- IP
- ID de evento

6. Requerimientos algorítmicos (CRÍTICOS)

6.1 Implementación de BST

Debe incluir:

- inserción
- búsqueda
- recorrido

6.2 Implementación de AVL (OBLIGATORIO)

Debe implementarse desde cero:

- cálculo de altura
- factor de balance

- rotaciones:
    - simple izquierda
    - simple derecha
    - doble izquierda-derecha
    - doble derecha-izquierda

No se permite usar librerías externas.

6.3 Control de balanceo

El sistema debe:

- detectar desbalance
- aplicar rotaciones automáticamente
- mantener la propiedad de BST

6.4 Simulación de escenarios

El estudiante debe ejecutar:

1. Inserción ordenada → demostrar degradación en BST
2. Inserción aleatoria → comparar comportamiento
3. Inserción masiva → evaluar escalabilidad

7. Análisis experimental

El estudiante debe medir:

- tiempo de inserción
- tiempo de búsqueda
- altura del árbol

Y comparar:

- BST vs AVL

👉 Debe evidenciar que AVL mantiene O(log n) mientras BST puede degradarse

8. Extensión (nivel proyecto)

El estudiante debe responder:

👉 ¿Cómo integrarías esta estructura en un sistema real de ciberseguridad?

Ejemplos esperados:

- indexación de logs
- listas negras dinámicas
- correlación de eventos

9. Análisis técnico obligatorio

Responde:

1. ¿Por qué el BST falla en escenarios reales?
2. ¿Qué garantiza el AVL que el BST no puede?
3. ¿Cuál es el costo del balanceo?
4. ¿En qué escenario AVL NO sería la mejor opción?
5. ¿Cómo afectaría esto a un SOC en tiempo real?

10. Uso de IA

Permitido solo para:

- generación de datos
- apoyo conceptual

Prohibido delegar:

- implementación AVL
- lógica de rotaciones

11. Declaración obligatoria de IA

Debe incluir en anexos:

- todos los prompts
- respuestas relevantes
- propósito

12. Requisitos técnicos

- Google Colab obligatorio
- código modular y comentado
- implementación propia

- análisis Big-O
- experimentación real

13. Entrega

PDF exportado desde Colab con:

- implementación
- resultados
- análisis
- conclusiones
- anexos

14. Rúbrica (100 puntos)

Criterio	**Puntaje**
Implementación BST	10
Implementación AVL	25
Correcto uso de rotaciones	15
Simulación de escenarios	10
Comparación experimental	10
Análisis de complejidad	10
Justificación técnica	10
Aplicación en ciberseguridad	5
Presentación	5

15. Evidencia de logro

El estudiante demuestra que:

- sabe implementar estructuras avanzadas
- entiende el impacto del balanceo
- puede diseñar soluciones eficientes
- razona como ingeniero