Proyecto Análisis y Diseño de Algoritmos

# 1. Introducción

Este proyecto busca integrar conocimientos avanzados en análisis algorítmico, técnicas de diseño y tecnologías emergentes, para construir un sistema que, a partir de un algoritmo escrito en pseudocódigo, determine automáticamente su complejidad computacional. El trabajo será desarrollado por parejas de estudiantes y tendrá como eje central la implementación de un Analizador de Complejidades asistido por Modelos de Lenguaje de Gran Escala (LLMs).

# 2. Objetivo General

Diseñar e implementar un sistema inteligente que analice algoritmos escritos en pseudocódigo, con el fin de determinar su complejidad en notación O (peor caso), Ω (mejor caso), Θ (caso promedio) y cotas fuertes, utilizando técnicas avanzadas de diseño algorítmico e integrando fundamentos teóricos y prácticos de los Modelos de Lenguaje (LLMs).

# 3. Descripción del Proyecto

El sistema debe ser capaz de interpretar estructuras algorítmicas como ciclos, condicionales, recursiones, llamadas a procedimientos, estructuras de datos (vectores, objetos, grafos, etc‥; Se definirán más adelante), y producir como salida un análisis detallado de su complejidad computacional. Este análisis debe incluir razonamientos y cálculos que evidencien el uso de técnicas como árboles de recursión, relaciones de recurrencia, programación dinámica, heurísticas, voraces, entre otros.

Una parte obligatoria del proyecto será la incorporación de modelos de lenguaje (LLMs), cuya función será asistir en:

* El sistema puede aceptar descripciones en lenguaje natural del algoritmo y usar un LLM para traducirlo a pseudocódigo estructurado.
* Llevar el problema a un análisis estructurado por cada paso en el proceso de razonamiento.
* Clasificación de patrones algorítmicos conocidos.
* Verificación o validación del análisis generado por el sistema mediante representación matemática.
* Diagramas de representación de seguimiento de la ejecución del pseudocódigo.
* Apoyo a la documentación explicativa del análisis realizado y coste por cada instrucción del análisis (microsegundos y tokens por llamado).

# 4. Entregables

* Informe técnico con el análisis del sistema desarrollado, metodología utilizada, técnicas aplicadas y ejemplos resueltos.
* Análisis completo del algoritmo del propio Analizador de Funciones de eficiencia (Mejor caso, peor caso y caso promedio según aplica) y por cada uno su Complejidades en notación O, Ω y Θ.
* Código fuente documentado, modularizado y funcional.
* Conjunto de pruebas con múltiples algoritmos de entrada dados (mínimo 10).
* Recurso explicativo (video/animación/presentación).

# 5. Criterios de Evaluación

|  |  |
| --- | --- |
| **Criterio** | **Ponderación** |
| Correcto análisis de complejidad (O, Ω, Θ) | 60% |
| Diagramas de seguimiento del pseudocódigo | 15% |
| Aplicación de técnicas algorítmicas avanzadas | 15% |
| Informe técnico y recursos | 5% |
| Cobertura de pruebas y validación | 5% |

# 6. Observaciones Finales

El proyecto representa una oportunidad para que los estudiantes integren conocimientos de análisis de algoritmos con nuevas tecnologías basadas en inteligencia artificial. Se espera un producto robusto, técnicamente sólido, y que explore con creatividad el uso de herramientas de soporte como los LLMs para el análisis algorítmico.

Propuesta de uso técnico de LLMs (Gemini/ChatGPT/Anthropic o similares):

* **Asistencia en parsing:** usar un LLM (vía API o manualmente) para sugerir la estructura lógica del algoritmo.
* **Comparación de resultados:** el sistema del estudiante produce una complejidad, y un LLM también, para ver concordancia o analizar divergencias.
* **Entrenamiento de un pequeño modelo de clasificación de estructuras algorítmicas**, usando ejemplos y GPT como “generador” de dataset.

Esto no es obligatorio, pero podría dar **crédito adicional**.