	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorios de docencia

Laboratorio de Computación Salas A y B

Profesor(a): M.C. Jorge Alberto Solano Galvez

Asignatura: Estructura de Datos y Algoritmo II

Grupo: 4

No de Práctica(s): 4 - Algoritmos de Búsqueda. Parte 1

Integrante(s): González Barragán Abraham Elienai

No. de lista o brigada:

Semestre: 2026-1

Fecha de entrega: 12 de septiembre de 2025

Observaciones:

CALIFICACIÓN: _____

1. Objetivo

Identificar el comportamiento y características de algunos algoritmos de búsqueda por comparación de llaves.

2. Descripción de Actividades

- Implementar **búsqueda lineal** en lenguaje Python tanto **iteraiva** como **recursiva** para encontrar un nodo.
- Obtener la **complejidad algorítmica teórica** para cada implementación (búsqueda secuencial y búsqueda binari).
- Obtener el **comportamiento empírico** de los algoritmos implementados para el mejor, el peor y el caso promedio (búsqueda secuencial y búsqueda).

3. Resultados Obtenidos

El código fuente y las gráficas obtenidas se encuentran en el archivo "practica4.ipynb".

4. Conclusiones

Esta práctica me dejó una comprensión más detallada de los métodos de búsqueda, lineal y la binaria, y la importancia de analizar la complejidad de los algoritmos. A través de la implementación de la búsqueda lineal, tanto de manera iterativa como recursiva, entendí que, aunque es un enfoque sencillo y funcional, su eficiencia disminuye drásticamente a medida que el conjunto de datos crece, con una complejidad de tiempo de $O(n)$ en el peor de los casos.

Por otro lado, la búsqueda binaria me mostró un enfoque más estratégico, al trabajar con un array ordenado, este algoritmo me permitió encontrar el elemento deseado de manera mucho más rápida, su complejidad de tiempo de $O(logn)$ es una clara demostración de cómo el diseño de un algoritmo puede impactar radicalmente su rendimiento, dividiendo el problema en cada paso para llegar a la solución.

Finalmente, el análisis de complejidad fue lo más importante, si bien comprendí correctamente que la complejidad temporal del algoritmo de merge sort es $O(nlogn)$, mi principal aprendizaje de esta práctica fue haber analizado incorrectamente su complejidad espacial. Ahora sé que el merge sort tiene una complejidad espacial de $O(nlogn)$, ya que requiere un espacio adicional para realizar la fusión de las sublistas, lo cual es un detalle fundamental para entender completamente su funcionamiento y recursos necesarios. Esta práctica me enseñó la importancia de considerar tanto el tiempo como el espacio al evaluar la eficiencia de cualquier algoritmo.