

# INFORME SOBRE LA EJECUCIÓN DE LOS ALGORITMOS DE ORDENAMIENTO

## *Equipo utilizado:*

Laptop: MSI PRESTIGE A4/ Intel Core i5-10210U @ 1.60–4.20 GHz

Memoria RAM: 16 GB

Sistema Operativo: Windows 10 / 64 bits

Python: ejecución local, sin librerías de optimización (solo time.perf\_counter)

## *Tamaños utilizados:*

Los tamaños elegidos para las pruebas fueron:

- 100 elementos (entrada muy pequeña)
- 50,000 elementos (entrada mediana-grande)
- 200,000 elementos (entrada muy grande)

## *Sobre las repeticiones:*

La consigna decía correr cada algoritmo 3 veces para cada tamaño.

Se hicieron todas las repeticiones completas para todos los algoritmos con  $n=100$  y  $n=50,000$ .

En los tamaños de 200,000 elementos, algunos algoritmos no completaron las 3 repeticiones debido al tiempo extremadamente alto que requieren. Esto no es un error del código ni del equipo, sino una consecuencia directa del costo computacional de los algoritmos  $O(n^2)$ .

## *Casos donde no fue posible completar las 3 repeticiones:*

### 1. Selection Sort (200,000)

- Se completaron 2 repeticiones.
- Cada repetición tardó aproximadamente 1,020,000 ms ( $\approx 17$  minutos).
- La tercera repetición no se completó para evitar sobrecalentamiento prolongado.

### 2. Insertion Sort (200,000)

- Se completaron 2 repeticiones.
- Cada repetición tardó entre 1,130,000 y 1,150,000 ms ( $\approx 18$ –19 minutos).
- Igual que en Selection, se detuvo antes de la repetición 3 por temperatura y tiempo.

### 3. Bubble Sort (200,000)

- Se completaron 2 repeticiones.
- Tiempos obtenidos: 2,734,892.517 ms y 2,689,441.900 ms ( $\approx 45$  minutos cada una).
- La tercera repetición no se realizó porque Bubble Sort en 200,000 elementos puede requerir más de una hora por repetición y provoca que el procesador permanezca a 100% durante tiempos prolongados.

Este comportamiento es completamente esperado y se documenta como parte del análisis.

### ***Conclusiones importantes basadas en los datos:***

- Los algoritmos  $O(n \log n)$  (Quick Sort, Heap Sort, Merge Sort) escalan de forma eficiente incluso hasta 200,000 elementos, con tiempos por debajo de 1–1.5 segundos.
- Heap Sort presentó valores más altos comparados con Quick y Merge debido a su mayor constante oculta, pero aun así fue estable y rápido.
- Los algoritmos  $O(n^2)$  muestran un crecimiento explosivo al pasar de 50,000 a 200,000 elementos:
  - Selection Sort pasó de ~53,000 ms a más de 1,020,000 ms.
  - Insertion Sort pasó de ~55,000 ms a más de 1,130,000 ms.
  - Bubble Sort pasó de ~127,000 ms a casi 2,700,000 ms.
- Bubble Sort es el peor escalado de todos; los resultados obtenidos son consistentes con su complejidad teórica (~20 mil millones de comparaciones en  $n=200,000$ ).
- A nivel térmico, el procesador alcanzó cargas del 100% en las pruebas  $O(n^2)$ , provocando throttling (reducción automática de frecuencia). Esto aumenta aún más el tiempo por repetición, lo que explica las variaciones obtenidas.
- La imposibilidad de completar 3 repeticiones en algunos casos no invalida el experimento: más bien demuestra empíricamente las limitaciones prácticas de los algoritmos ineficientes cuando escalan a tamaños altos.