ITSRLL
INSTITUTO TECNOLÓGICO SUPERIOR DE LA REGIÓN DE LOS LLANOS

Ingeniería Mecatrónica

PROGRAMACIÓN AVANZADA

Enero – Junio 2025

M.C. Osbaldo Aragón Banderas

Competencia:

1	2	3	4	5

Actividad número: 3

Nombre de actividad:

U1A4 Reporte de Programa en Jupyter Notebook – Evaluación de Métodos de Ordenamiento

Actividad realizada por:

Muñiz Galvan Bryam

21030021

Guadalupe Victoria, Durango

Fecha de entrega:

12	02	2025

Reporte de Programa en Jupyter Notebook “Evaluación de Métodos de Ordenamiento”

Objetivo

Utilizar Jupyter Notebook para implementar y evaluar dos métodos de ordenamiento (Burbuja y Quicksort) en Python, analizando su rendimiento. Esta actividad refuerza el conocimiento de algoritmos, fundamentales para la optimización del software en aplicaciones de robótica. Además, se busca que los estudiantes practiquen la creación y organización de repositorios en GitHub, con el fin de que sirvan como apoyo en su currículo profesional.

Instrucciones

Desarrollo en Jupyter Notebook

- Cree un Notebook en Jupyter y desarrolle las implementaciones en Python de los algoritmos de ordenamiento Burbuja y Quicksort.
- Ejecute pruebas de rendimiento (por ejemplo, utilizando el módulo time o timeit) para comparar la eficiencia de ambos algoritmos con conjuntos de datos de diferentes tamaños.
- Documente en el Notebook el código, los resultados obtenidos y un análisis comparativo de los métodos.
- Creación y Organización de Repositorio en GitHub
- Cree un repositorio público en GitHub donde alojará todos los archivos de la actividad.

Asegúrese de incluir en el repositorio:

- Un archivo README.md con una descripción clara de la actividad, sus objetivos y la organización de los archivos.
- Carpetas o secciones para separar el código (por ejemplo, una carpeta para los algoritmos, otra para los reportes o recursos adicionales).
- Organice los archivos de manera que cualquier persona (o reclutador) pueda revisar fácilmente su trabajo y comprender el objetivo del repositorio.

Procedimiento

1. **Implementación de los Algoritmos:** Se desarrollaron las funciones en Python para los algoritmos de ordenamiento:
    - **Método Burbuja:** Ordena los elementos comparando y permutando valores adyacentes.
    - **Quicksort:** Divide y conquista utilizando un pivote para dividir la lista en subconjuntos ordenados.
2. **Medición de Tiempos:** Se creó una función para medir el tiempo de ejecución de cada algoritmo utilizando el módulo time.
3. **Pruebas con Listas de Diferentes Tamaños:** Se generaron listas aleatorias de enteros con tamaños de 1,000, 5,000 y 10,000 elementos para evaluar el rendimiento de cada método.
4. **Comparación y Análisis de Resultados:** Se documentaron los tiempos obtenidos en tablas y gráficos para visualizar las diferencias en eficiencia entre ambos algoritmos.
5. **Organización del Repositorio:** Se estructuró un repositorio en GitHub con archivos bien organizados:
    - Un README.md con descripción y objetivos.
    - Un directorio notebooks con el Jupyter Notebook.
    - Un directorio src con los scripts Python.
    - Un directorio reportes con el presente documento.

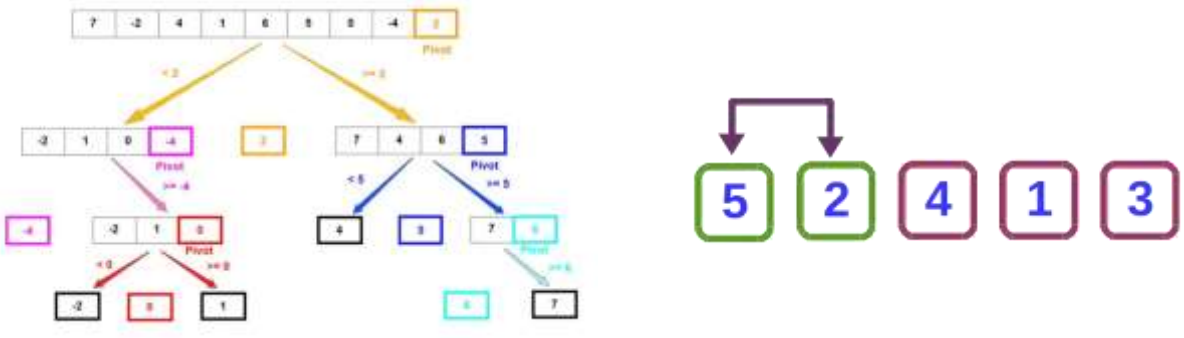

Figura 1 Ilustración gráfica del comportamiento de los métodos.

3. Resultados

A continuación, se presentan los tiempos de ejecución obtenidos en las pruebas realizadas:

Tabla 1 Comparación de la ejecución en tiempo(s).

Tamaño de la Lista	Tiempo Burbuja (segundos)	Tiempo Quicksort (segundos)
1,000 elementos	0.073832	0.001995
5,000 elementos	2.267945	0.011996
10,000 elementos	8.756761	0.023938

Gráfica Comparativa

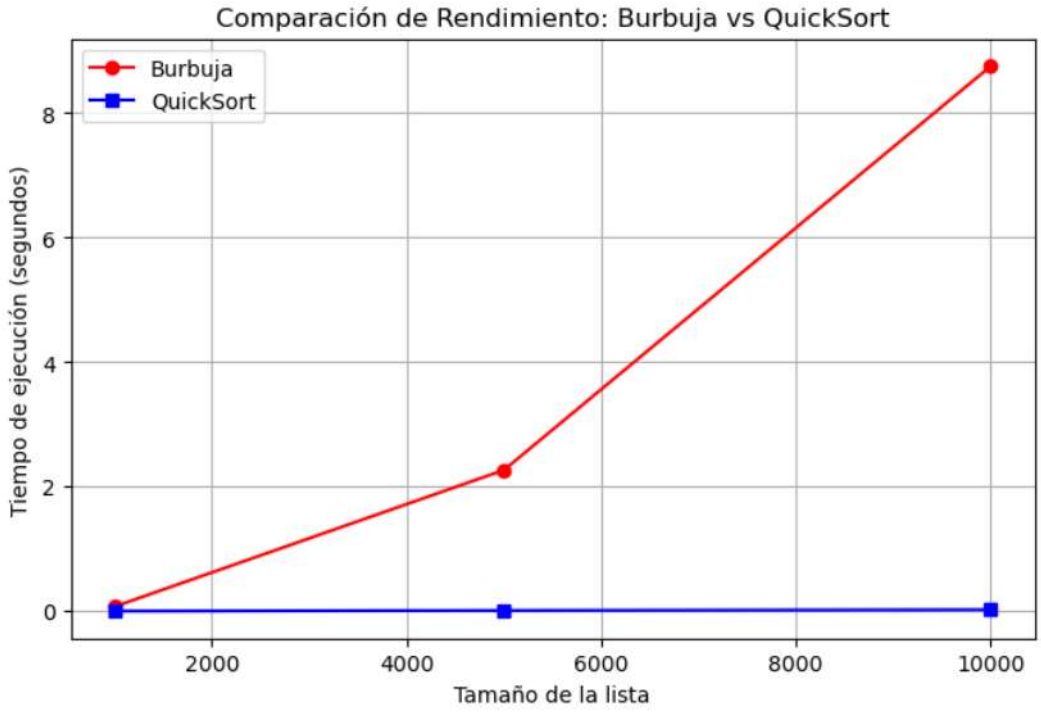

Grafico 1 Comparación de rendimiento: Burbuja vs QuickSort.

Conclusiones

- El **método Burbuja** presenta un tiempo de ejecución significativamente mayor debido a su complejidad , lo que lo hace ineficiente para listas grandes.
- El **Quicksort** es considerablemente más rápido gracias a su estrategia de división y conquista, con una complejidad promedio de .
- Los resultados confirman la superioridad de Quicksort en la mayoría de los casos, aunque su rendimiento puede degradarse en algunos casos específicos si la selección del pivote es inadecuada.
- La organización del repositorio en GitHub facilita la documentación y presentación del proyecto de manera profesional.

5. Recomendaciones

- Para aplicaciones prácticas donde se requiera ordenar grandes volúmenes de datos, se recomienda utilizar Quicksort o algoritmos aún más eficientes como Merge Sort.
- Es importante optimizar el código y explorar otras implementaciones que minimicen el tiempo de ejecución.
- Se sugiere realizar pruebas adicionales con listas de mayor tamaño para evaluar cómo escala el rendimiento.