1. Disponer de un ambiente ya sea virtualizado o local con Linux, cualquier distribución.
2. Programar 3 algoritmos de ordenamiento utilizando el Lenguaje C.
   1. Quick Sort.
   2. Bubble Sort.
   3. Insertion Sort.
3. Realizar la generación de un archivo de 100 MB con números aleatorios entre 1 y 100.000, estos números no deben ser impresos por pantalla y deben quedar en un archivo de texto plano.
4. Posterior a la generación se deben leer los números para ser ordenados en un nuevo archivo o bien pueden ir generando los números para ser almacenados directamente en un arreglo y luego guardar el archivo generados, con los números ordenados por cada algoritmo.
5. Lleve a cabo el mismo proceso, pero ahora utilizando el lenguaje Java.
6. Tomar los tiempos de ejecución de cada algoritmo para cada lenguaje, utilizando el comando time.
7. Al mismo tiempo que ejecuta el comando time, evaluar en otras dos terminales utilizando los comandos vmstat e iostat.
8. Identifique las diferencias entre ambos códigos y lenguajes, Ejemplo.

|  |  |  |  |
| --- | --- | --- | --- |
| Algoritmo | Tiempo C | Tiempo Java | Observación |
| QSort |  |  |  |
| Bubble Sort |  |  |  |
| Insertion Sort |  |  |  |

1. Grafique los resultados para cada algoritmo y por comando, ejemplo Iostat y vmstat. Puede investigar otros comandos que puedan servir de ayuda.
2. ¿Por qué se produce una diferencia entre ambos algoritmos/lenguaje?
3. ¿Cómo influye los discos utilizados en el equipo?
4. ¿Cuál es la principal diferencia entre Java y C?
5. ¿Qué podría mejorar el rendimiento de estos algoritmos?
6. ¿Cómo se comporta el I/O y el acceso a discos y/o memoria al momento de realizar la ejecución?
7. Si uno de los dos algoritmos funciona más eficientemente, ¿cuál es el recurso que esta afectando al otro algoritmo? ¿Cómo podría mejorarlo? ¿Cómo afectaría la aceleración con un sustituto?