• ¿El comportamiento con relación al orden de crecimiento temporal de los algoritmos es acorde a lo enunciado teóricamente? :

Sí, se tiene en cuenta el orden de crecimiento de cada algoritmo con respecto al tiempo,

-El ordenamiento merge tiene un crecimiento Nlog(N) que lo hace linearítmico.

-El ordenamiento quick tiene un crecimiento N^2 que lo hace cuadrático.

-El ordenamiento shell tiene un un crecimiento de N^3/2 que lo puede hacer más eficiente que el ordenamiento quick, sin embargo detectamos más efectividad en el ordenamiento quick basados en los resultados obtenidos de nuestras pruebas del proyecto. Teniendo en cuenta esto podemos decir que en las pruebas realizadas se detectó una incongruencia entre la práctica y la teoría.

-El ordenamiento Insertion posee una complejidad cuadrática N^2 tanto en el mejor como en el peor de los casos. Esto hace que sin importar la cantidad de datos y cómo están ordenados inicialmente siempre se va a recorrer la lista completa, haciéndolo el más demorado de los ordenamientos.

• ¿Existe alguna diferencia entre los resultados obtenidos al ejecutar las pruebas en diferentes máquinas?

Sí, principalmente tanto de procesador como de memoria

• De existir diferencias, ¿a qué creen que se deben?:

La cantidad de hilos del procesador, la frecuencia a la que trabaja, los núcleos que este posee, al igual que lo moderno y nuevo que sea este, debido a que un procesador de última generación no trabaja al mismo ritmo que uno que ya tiene bastante tiempo en el mercado. También, se debe tener en cuenta la forma en cómo los distintos tipos de datos usan la memoria y como buscan su máxima eficiencia, aunque depende de la cantidad que se disponga en el ordenador debido a que uno llegará a su máximo de capacidad más rapido que otro.

• ¿Cuál Estructura de Datos (ARRAY\_LIST o SINGLE\_LINKED) funciona generalmente mejor si solo se tiene en cuenta los tiempos de ejecución de los algoritmos?

Array\_List, debido a que este no se maneja por nodos como lo hace el Single\_Linked y todos los datos están linealmente agrupados, esto permite que se pueda acceder a cada uno de los datos por su posición única y evitar perder tiempo en el proceso, además el Single\_linked, se maneja por apuntadores de cada uno de los datos de la lista, cosa que hace que se demore más el proceso

• Teniendo en cuenta las pruebas de tiempo de ejecución reportadas por los algoritmos de ordenamiento probados (iterativos y recursivos), proponga un listado de estos ordenarlos de menor a mayor teniendo en cuenta el tiempo de ejecución que toma ordenar las obras de arte.

El más rápido fue el Mergesort, seguido del Quicksort, después el Shellsort y finalmente el Insertionsort