OBSERVACIONES DE LA PRACTICA

Estudiante 1 Maria Alejandra Moreno Bustillo Cod 202021603
Estudiante 2 Juliana Delgadillo Cheyne Cod 202020986

Ambientes de pruebas

	Máquina 1	Máquina 2
Procesadores	AMD Ryzen 7 3700U with Radeon Vega Mobile Gfx 2.30 GHz	1.6 GHz Dual-Core Intel Core i5
Memoria RAM (GB)	12,0 GB (9,95 GB usable)	8 GB 1600 MHz DDR3
Sistema Operativo	Windows 10 x64bits	MacOS Big Sur - Versión 11.5.2

Tabla 1. Especificaciones de las máquinas para ejecutar las pruebas de rendimiento.

Maquina 1

Resultados

Cada método se probó 4 veces y el promedio de los tiempos se encuentra documentado en la tabla

Porcentaje de la muestra [pct]	Tamaño de la muestra (ARRAYLIST)	Insertion Sort [ms]	Shell Sort [ms]	Quick Sort [ms]	Merge Sort [ms]
0.50%	Archivo -small (768)	23.43	27.34	23.43	27.34
10.00%	Archivo -10pct (15008)	429.69	1171.87	4304.69	570.32

Tabla 2. Comparación de tiempos de ejecución para los ordenamientos en la representación arreglo.

Porcentaje de la muestra [pct]	Tamaño de la muestra (LINKED_LIST)	Insertion Sort [ms]	Shell Sort [ms]	Quick Sort [ms]	Merge Sort [ms]
0.50%	Archivo -small (768)	1796.88	1527.35	1410.155	171.88
10.00%	Archivo -10pct (15008)	342281.25	1019484.38	2479156.25	68179.59

Tabla 3. Comparación de tiempos de ejecución para los ordenamientos en la representación lista enlazada.

Algoritmo	Arreglo (ARRAYLIST)	Lista enlazada (LINKED_LIST)
Insertion Sort	Más eficiente de todos	Segundo más eficiente
Shell Sort	Tercero más eficiente de todos	Tercero más eficiente de todos
Merge Sort	Segundo más eficiente	Más eficiente de todos
Quick Sort	Menos eficiente de todos	Menos eficiente de todos

Tabla 4. Comparación de eficiencia de acuerdo con los algoritmos de ordenamientos y estructuras de datos utilizadas.

Maquina 2

Resultados

Porcentaje de la muestra [pct]	Tamaño de la muestra (ARRAYLIST)	Insertion Sort [ms]	Shell Sort [ms]	Quick Sort [ms]	Merge Sort [ms]
0.50%	Archivo -small (768)	47.06	46.06	40.56	35.59
10.00%	Archivo -10pct (15008)	678.81	1348.47	4335.84	786.48

Tabla 5. Comparación de tiempos de ejecución para los ordenamientos en la representación arreglo.

Porcentaje de la muestra [pct]	Tamaño de la muestra (LINKED_LIST)	Insertion Sort [ms]	Shell Sort [ms]	Quick Sort [ms]	Merge Sort [ms]
0.50%	Archivo -small (768)	2036.66	1722.91	1531.88	207.34
10.00%	Archivo -10pct (15008)	660471.26	1692633.83	3627835.91	84896.94

Tabla 6. Comparación de tiempos de ejecución para los ordenamientos en la representación lista enlazada.

Algoritmo	Arreglo (ARRAYLIST)	Lista enlazada (LINKED_LIST)
Insertion Sort	Más eficiente de todos	Segundo más eficiente
Shell Sort	Tercero más eficiente de todos	Tercero más eficiente de todos
Merge Sort	Segundo más eficiente	Más eficiente de todos
Quick Sort	Menos eficiente de todos	Menos eficiente de todos

Tabla 7. Comparación de eficiencia de acuerdo con los algoritmos de ordenamientos y estructuras de datos utilizadas.

Preguntas de análisis

1) ¿El comportamiento de los algoritmos es acorde a lo enunciado teóricamente?

De acuerdo a lo enunciado de forma teórica y los resultados obtenidos al correr el programa en dos computadores con capacidades de rendimiento distintas podemos concluir que efectivamente aquellos algoritmos con los valores más reducidos en cuanto a θ son aquellos que gastan menos tiempo en ejecutar el programa, por ejemplo se observa la efectividad de Merge (recuersivo) el cual tiene un costo de O(nlog(n)). Así mismo vemos el comportamiento de los algoritmos iterativos donde Insertion con un peor caso de O(n^2) tiene un buen rendimiento cuando se utilizan arreglos mientras que el algoritmo Shell con O(n^3/2) siempre queda de tercero tanto para arreglos como para Single Linked. A pesar de todo lo anterior, nos dimos cuenta que el algoritmo quick sort (recursivo) tomo demasiado tiempo para realizar la ejecución lo cual nos llamo la atención dado que su mejor caso es de O(nlog(n)), sin embargo consideramos que este comportamiento ocurrió debido a que el algoritmo tiende a tomar mas tiempo cuando hay muchos elementos repetidos o la lista se encuentra parcialmente ordenada. Finalmente, cabe resaltar que se logro comprobar que las listas enlazadas funcionan mucho mejor con una algoritmo recursivo lo cual fue demostrado con las prueba realizadas para merge sort, en ambas pruebas, merge sort salió como el más rápido para ordenar datos en listas encadenadas.

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

Existe una gran diferencia al ejecutar las pruebas de tiempos entre las dos máquinas pues es evidente el aumento de tiempo (ms) que existe entre las ejecuciones de la máquina 1 y la máquina 2. En todas las pruebas realizadas se calcula un tiempo significativamente mayor en la máquina 2 que en algunos casos incluso llega a duplicar el tiempo medido para la máquina 1.

3) **De existir diferencias, ¿a qué creen que se deben?**

Las diferencias de tiempo que se evidencian en las pruebas ocurren debido al rendimiento y capacidades de hardware con las que cuenta cada máquina. Es posible realizar esta afirmación, porque durante la ejecución del programa ambos computadores se encontraban con todos los aplicativos cerrados, de tal forma que ninguna ventana abierta afectara los resultados obtenidos, permitiendo que el dispositivo solo se encargara de ejecutar el programa. Esto nos lleva a concluir que las diferencias se deben principalmente al hecho que la máquina 1 tiene mayor memoria RAM y un procesador con mayor potencia, lo que le permite realizar los ordenamientos en menor tiempo que la maquina 2. Esto además se puede evidenciar en las velocidades de reloj que puede alcanzar cada procesador, la máquina 2 puede alcanzar hasta 1.6 GHz, mientras que la maquina 1 alcanza hasta 2.3 GHz, lo cual demuestra que puede hacer los procesos mucho más rápidos.

4) **¿Cuál Estructura de Datos funciona mejor si solo se tiene en cuenta los tiempos de ejecución de los algoritmos?**

Con base a lo evidenciado en los resultados de las pruebas, es posible concluir que los Arreglos son la estructura de datos que mejores tiempos obtiene para ejecutar el programa. Como se observa en las tablas de las dos máquinas, para los cuatro diferentes algoritmos de ordenamiento se obtuvo un menor tiempo cuando los datos ingresados se cargaban como un Array List en comparación con la estructura Single Linked.

5) **Teniendo en cuenta las pruebas de tiempo de ejecución por todos los algoritmos de ordenamiento estudiados (iterativos y recursivos), proponga un ranking de los mismo de mayor eficiencia a menor eficiencia en tiempo para ordenar la mayor cantidad de obras de arte.**

Puesto	Algoritmo de Ordenamiento	Tipo de Algoritmo
1	Insertion Sort	Iterativo
2	Merge Sort	Recursivo
3	Shell Sort	Iterativo
4	Quick sort	Recursivo

Cabe resaltar que Insertion y Merge están en la disputa del primer puesto, pues cada uno es primero en alguno de los TADlista, insertion en Array list y Merge en Single linked.