OBSERVACIONES DE LA PRÁCTICA

Santiago Rojas Cod 202420928
Luis Alejandro Rodriguez Cod 202321287
Estudiante 3 Cod XXXX

Ambientes de pruebas

	Computador 1
Procesador	AMD Ryzen 5 4600 H
Memoria RAM (GB)	8
Sistema Operativo	Windows 10

Tabla 1. Especificaciones del computador para ejecutar las pruebas de rendimiento.

Para realizar las siguientes pruebas, es importante que las ejecuten utilizando el archivo de datos **large**, ya que es con este conjunto donde realmente se puede observar el comportamiento y la eficiencia de los algoritmos en contextos más exigentes.

Ordenamientos Iterativos

Resultados para ordenamientos iterativos con Array List

Porcentaje de la muestra	Insertion Sort (Array List)	Selection Sort (Array List)	Shell Sort (Array List)
0.50%	**1.18**	**1.097**	**0.363**
5.00%	**68.452**	**123.032**	**7.667**
10.00%	**169.911**	**542.237**	**21.569**
20.00%	**836.292**	**2176.554**	**93.697**
30.00%	**2876.183**	**4916.033**	**179.273**
50.00%	**15971.908**	**11325.732**	**507.418**
80.00%	**16806.947**	**25704.664**	**1068.097**
100.00%	**23236.047**	**40619.149**	**1692.649**

Tabla 2. ***Resultados en computador 1 para ordenamientos iterativos con Array List.***

Resultados para ordenamientos iterativos con Linked List

Porcentaje de la muestra [pct]	Tamaño de la muestra (LINKED_LIST)	Insertion Sort Linked List	Selection Sort Linked List	Shell Sort Linked List
0.50%	50.00	5.458	16.262	2.573
5.00%	500.00	5741.644	8003.298	465.305
10.00%	1000.00	46589.084	38117.784	3587.294
20.00%	2000.00	375896.087	350901.418	17240.696
30.00%	3000.00	1271913.476	99999999	43328.545
50.00%	5000.00	99999999	99999999	162086.974
80.00%	8000.00	99999999	99999999	555812.567
100.00%	10000.00	99999999	99999999	99999999

Tabla 3. ***Resultados en computador 1 para ordenamientos iterativos con Single Linked List.***

Ordenamientos Recursivos

Resultados para ordenamientos recursivos con Array List

Merge Sort Array List	Quick Sort Array List
0.864	0.852
12.705	15.701
29.099	34.237
68.346	110.128
120.316	197.589
193.718	518.803
352.433	1179.056
400.831	1824.785

Tabla 4. ***Resultados en computador 1 para ordenamientos recursivos con Array List.***

Resultados para ordenamientos recursivos con Linked List

Merge Sort Linked List	Quick Sort Linked List
1.156	0.709
48.707	12.802
154.255	36.44
363.292	105.951
1740.411	194.321
3587.101	523.781
12169.523	1395.499
16046.59	2347.523

Tabla 5. ***Resultados en computador 1 para ordenamientos recursivos con Single Linked List.***

Una vez alla llenado l

Comparación de tiempo mejores algoritmos de ordenamiento

Complete esta sección únicamente cuando se le indique en las instrucciones. En este punto, deberá comparar el tiempo de ejecución entre el mejor algoritmo de ordenamiento recursivo y el mejor algoritmo de ordenamiento iterativo, los cuales debió haber identificado en el punto anterior.

Por favor, registre las mediciones en el espacio correspondiente de la tabla, especificando claramente:

- El nombre de cada algoritmo utilizado
- El tipo de estructura de datos empleada **(*ArrayList* o *SingleLinkedList*)**

Asegúrese de completar todos los campos solicitados en la tabla e incluir los nombres según corresponda, no es necesario que vuelva a correr las pruebas esta información la puede sacar de las tablas anteriores.

Computador 1

Porcentaje de la muestra [pct]	Tamaño de la muestra	Algoritmo iterativo (Array List:Shell Sort)	Algoritmo recursivo (Array List: merge sort)
0.50%	50.00	**0.363**	**0.864**
5.00%	500.00	**7.667**	**12.705**
10.00%	1000.00	**21.569**	**29.099**
20.00%	2000.00	**93.697**	**68.346**
30.00%	3000.00	**179.273**	**120.316**
50.00%	5000.00	**507.418**	**193.718**
80.00%	8000.00	**1068.097**	**352.433**
100.00%	10000.00	**1692.649**	**400.831**

Tabla 6. ***Resultados en computador 1 mejores algoritmos***

Preguntas de análisis parte 1

1. **¿Cómo varía el comportamiento de cada algoritmo con respecto al tamaño de los datos?**

   Para responder, agregue líneas de tendencia en las gráficas de las pestañas 04 a 08**.** Observe si el crecimiento del tiempo de ejecución es lineal, cuadrático o de otro tipo, y relacione ese patrón con la complejidad teórica esperada de cada algoritmo.

   RTA: Al ver las gráficas con las líneas de tendencia, Insertion y Selection Sort aumentan de forma cuadrática, Shell Sort crece menos con un comportamiento intermedio, y Merge y Quick Sort muestran un crecimiento logarítmico, lo que coincide con sus complejidades en la teoría.

2. **¿Qué diferencias observas en el rendimiento de un mismo algoritmo al utilizar ArrayList frente a SingleLinkedList?**

Analice esta comparación en las pestañas 04 a 08, donde se muestran ambos casos. Justifique su respuesta con base en las líneas de tendencia.

RTA: La comparación entre ArrayList y SingleLinkedList muestra que en ArrayList los tiempos son menores, mientras que en SingleLinkedList los tiempos se disparan, llegando incluso a valores de 9999 cuando los algoritmos, especialmente Insertion y Selection, tardaban más de 20 minutos en completarse. Esto se refleja en líneas de tendencia mucho más grandes.

3. **¿Qué algoritmo iterativo mostró el mejor comportamiento general en términos de tiempo y cantidad de datos en ambas estructuras de datos?**

   Justifique su elección comparando las gráficas (04 Insertion Sort), (05 Selection Sort) y (06 Shell Sort) y apoyándose en las gráficas de las pestañas 2 y 3.

RTA: Entre los algoritmos iterativos, Shell Sort fue el que mejor se comportó, con una línea de tendencia mucho más suave que la de Insertion y Selection.

4. **¿Qué algoritmo recursivo presentó el mejor desempeño general según los tiempos registrados y la forma de la línea de tendencia?**

   Justifique su respuesta basándose en las pestañas 07 (Merge Sort) y 08 (Quick Sort) y apoyándose en las gráficas de las pestañas 2 y 3.

   RTA: Entre los recursivos, Merge Sort mostró un mejor desempeño general, con una tendencia más estable y de menor crecimiento que Quick Sort.

5. **Con base en su análisis visual y los datos recolectados, ¿cuáles algoritmos seleccionan como los "mejores" en cada categoría (iterativo y recursivo)?**

   Indique también la estructura de datos en la que se desempeñó mejor cada uno.

   RTA: En conclusión, al comparar las gráficas y sus tendencias, el mejor algoritmo iterativo fue Shell Sort en ArrayList, mientras que el mejor recursivo fue Merge Sort en ArrayList, donde los demás algoritmos presentaron valores demasiado altos.

Preguntas de análisis parte 2

1. **¿Cuál de los dos algoritmos seleccionados como "mejores" muestra un comportamiento más eficiente a medida que crece el tamaño de la muestra?**

   Para responder, agregue líneas de tendencia en la gráfica de la pestaña 09-Bests, identifique cuál algoritmo crece más lentamente y relacione ese patrón con su complejidad teórica. Use esta comparación para justificar cuál es más escalable.

RTA: Entre los dos mejores, Merge Sort crece más lentamente en las gráficas, lo que refleja su complejidad O(n log n) y lo hace más escalable que Shell Sort.

2. **¿Qué estructura de datos resultó más favorable para cada algoritmo seleccionado como mejor?**

   Analice si la ventaja se mantiene constante en todos los tamaños o solo en ciertos rangos.

   RTA: Shell y Merge funcionaron mejor en ArrayList, ya que en SingleLinkedList los tiempos se dispararon y llegaron hasta más de 20 minutos en algunos casos; esta ventaja se mantuvo en todos los tamaños de muestra.

3. **¿Qué ventajas prácticas podría tener implementar el algoritmo ganador en aplicaciones reales donde se manejan grandes volúmenes de datos?**

   RTA: En la práctica, Merge Sort es útil en aplicaciones con grandes volúmenes de datos porque mantiene tiempos bajos y estables incluso cuando la muestra crece mucho.

4. **Si se presentara una lista parcialmente ordenada, ¿cambiaría su elección de algoritmo? Explique por qué, con base en el comportamiento observado.**

   RTA: Si la lista estuviera parcialmente ordenada, Shell Sort se vería más beneficiado, aunque Merge Sort seguiría siendo mejor por su crecimiento logarítmico.