OBSERVACIONES DE LA PRACTICA

Jose David Florez Ruiz Cod 202121507
Santiago Castro Duque Cod 202122158

	Máquina 1	Máquina 2
Procesadores	AMD Ryzen 5 4600H with Radeon Graphics 3.00 GHz	Intel i3 –7100U CPU @ 2.40GHz
Memoria RAM (GB)	16	8
Sistema Operativo	Windows 10	Windows 10

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

Maquina 1

Resultados

Porcentaje de la muestra [pct]	Tamaño de la muestra (ARRAY_LIST)	Insertion Sort [ms]	Selection Sort [ms]	Shell Sort [ms]	Quick Sort [ms]	Merge Sort [ms]
0.50%		2485.78	2456.92	24.92	71.65	112.25
5.00%		149571.55	146163.78	1131.16	908.62	816.78
10.00%		439645.12	430656.26	2308.49	1895.20	1441.19
20.00%				4685.04	3164.11	2414.51
30.00%				6036.72	4403.36	2994.14
50.00%				7133.88	5845.87	4430.47
80.00%				9336.49	8491.37	5104.90
100.00%				11516.27	8345.47	5535.08

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

Porcentaje de la muestra [pct]	Tamaño de la muestra (LINKED_LIST)	Insertion Sort [ms]	Selection Sort [ms]	Shell Sort [ms]	Quick Sort [ms]	Merge Sort [ms]
0.50%		2485.78	241130.11	10664.38	7495.88	9235.84
5.00%				1177257.49	182633.52	128548.29
10.00%						174265.7
20.00%						
30.00%						
50.00%						
80.00%						

100.00%

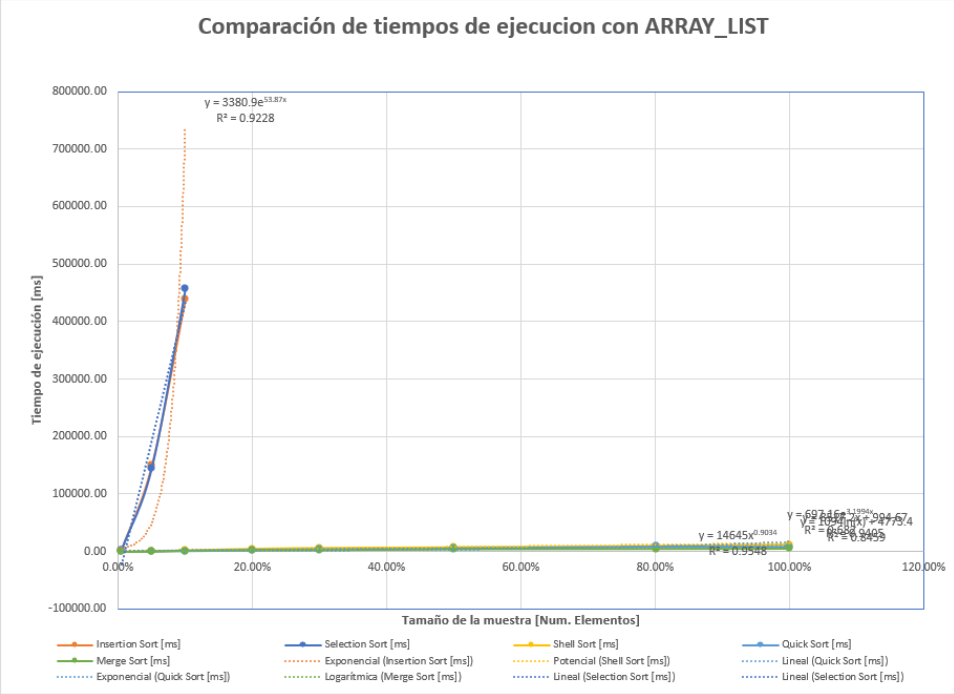
Tabla 3. Comparación de tiempos de ejecución para los ordenamientos iterativos en la representación lista enlazada.

Algoritmo	Arreglo (ARRAYLIST)	Lista enlazada (LINKED_LIST)
Merge sort	=>	
Quick sort	=>	

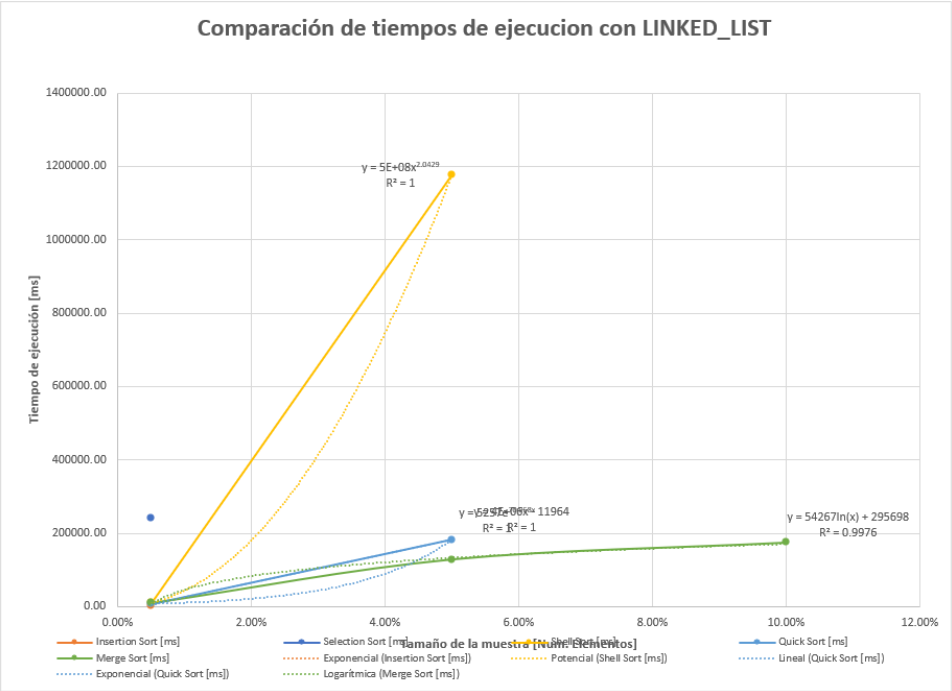
Tabla 4. Comparación de eficiencia de acuerdo con los algoritmos de ordenamientos y estructuras de datos utilizadas.

Graficas

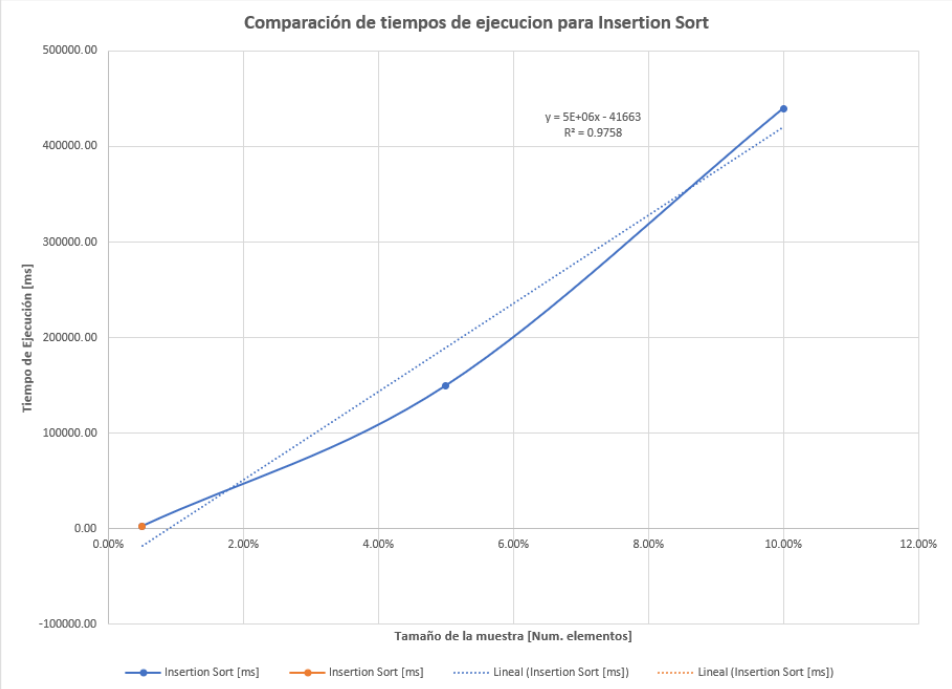
- Cinco gráficas generadas por los resultados de las pruebas de rendimiento en la **Maquina 1.**
  - Comparación de rendimiento ARRAYLIST.



  - Comparación de rendimiento LINKED_LIST.



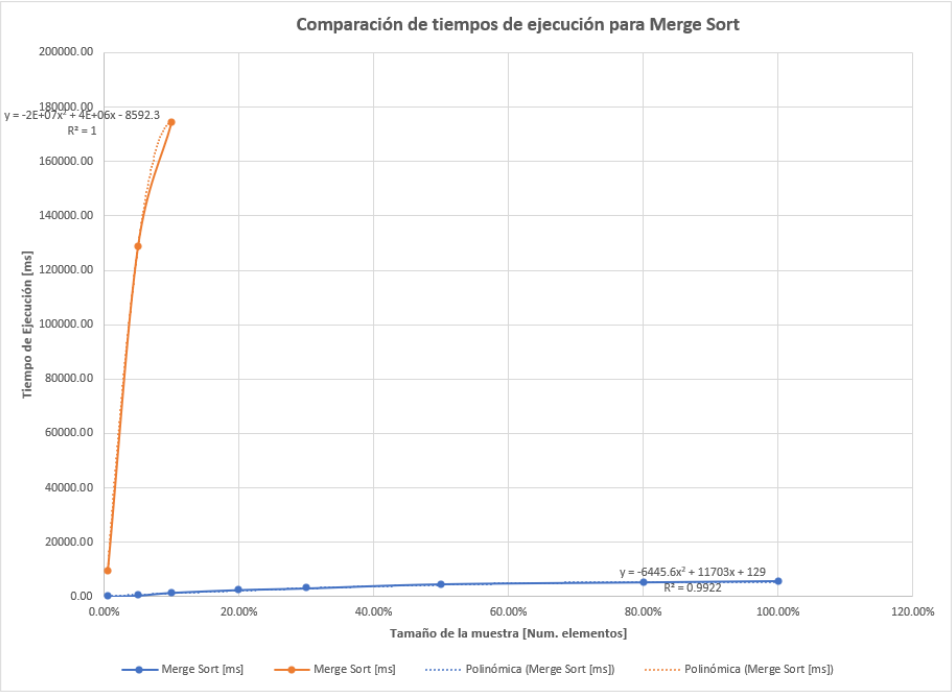
- Comparación de rendimiento para Insertion Sort.



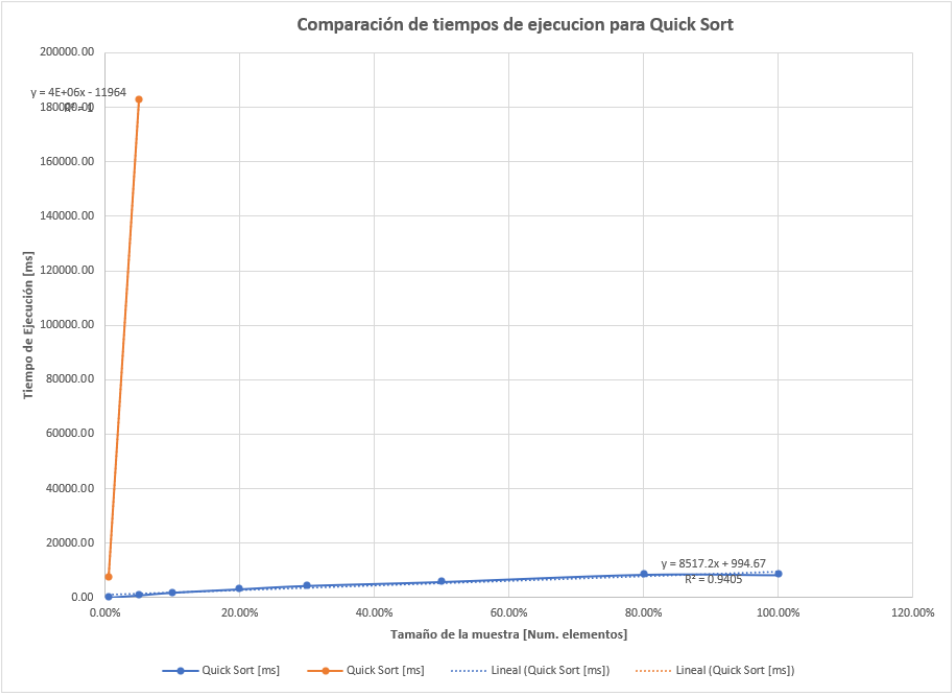
- Comparación de rendimiento para Selection Sort.

- Comparación de rendimiento para Shell Sort.

- Comparación de rendimiento para MergeSort.



- Comparación de rendimiento para QuickSort.



Maquina 2

Resultados

Porcentaje de la muestra [pct]	Tamaño de la muestra (ARRAY_LIST)	Insertion Sort [ms]	Selection Sort [ms]	Shell Sort [ms]	Quick Sort [ms]	Merge Sort [ms]
0.50%		1.89	328.59	13.40	18.17	19.59
5.00%		28207.42	27089.9	108.47	108.	106.91
10.00%		83450.11	84088.16	155.08	389.20	423.77
20.00%		286970.33	269045.38	295.44	976.02	814.89
30.00%		516443.66	516443.66	454.08	1456.48	1209.27
50.00%				4045.55	2437.11	2502.19
80.00%				6380.93	4171.90	2731.61
100.00%				8569.52	6070.13	4063.17

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

Porcentaje de la muestra [pct]	Tamaño de la muestra (LINKED_LIST)	Insertion Sort [ms]	Selection Sort [ms]	Shell Sort [ms]	Quick Sort [ms]	Merge Sort [ms]
0.50%	1000	17142.49	12647.4	248.84	276.75	264.10
5.00%	2000	12472770.14	7700229.11	176174.76	120408.50	10881.57
10.00%	4000			676356.36	386987.47	29155.48
20.00%	8000					117136.59
30.00%	16000					
50.00%	32000					
80.00%						
100.00%						

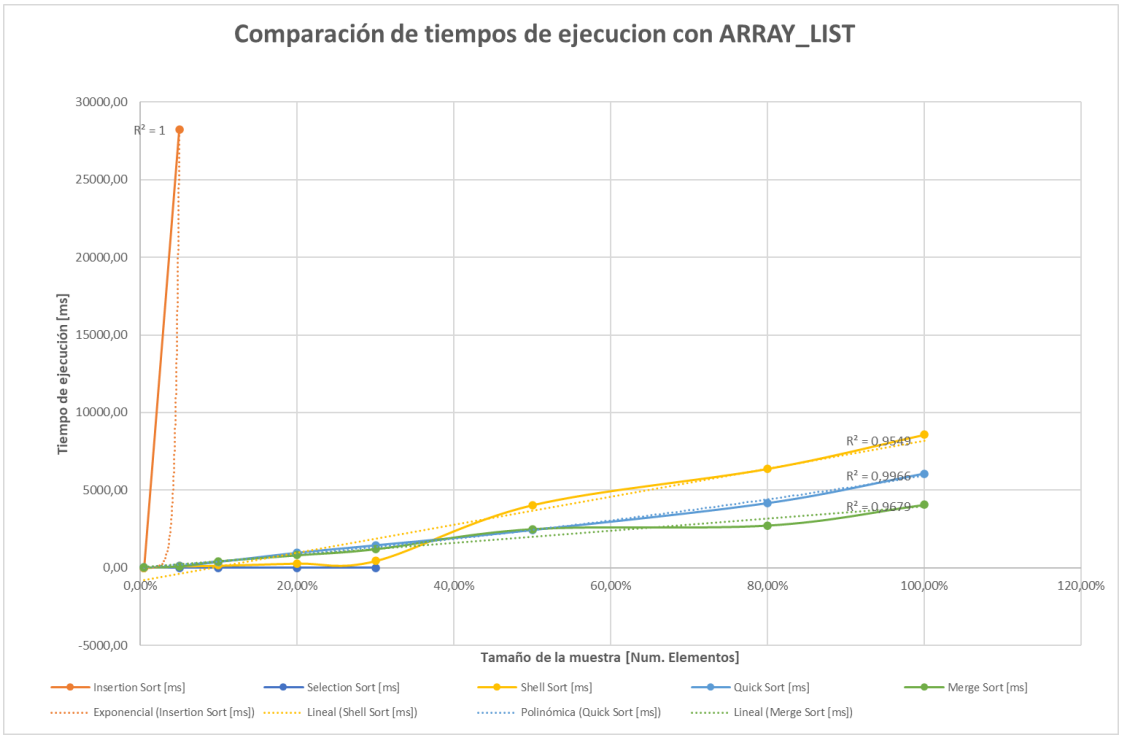
Tabla 6. Comparación de tiempos de ejecución para los ordenamientos iterativos en la representación lista enlazada.

Algoritmo	Arreglo (ARRAYLIST)	Lista enlazada (LINKED_LIST)
Merge sort	Más Eficiente	
Quick sort	Más Eficiente	

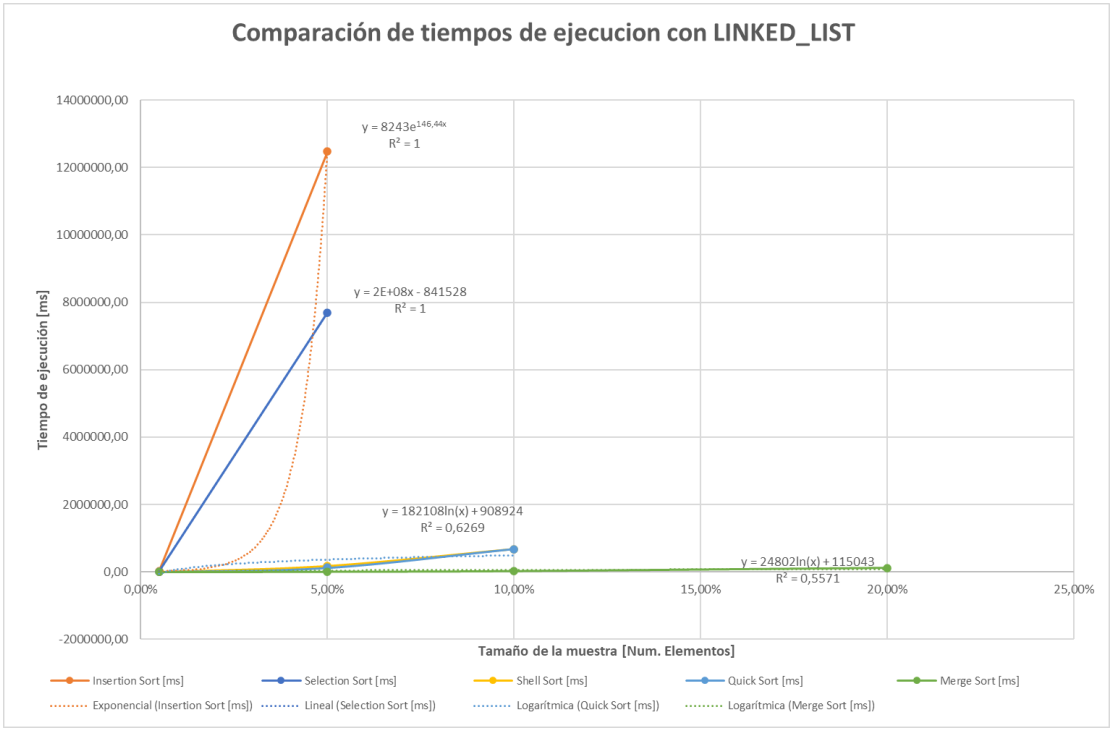
Tabla 7. Comparación de eficiencia de acuerdo con los algoritmos de ordenamientos y estructuras de datos utilizadas.

Graficas

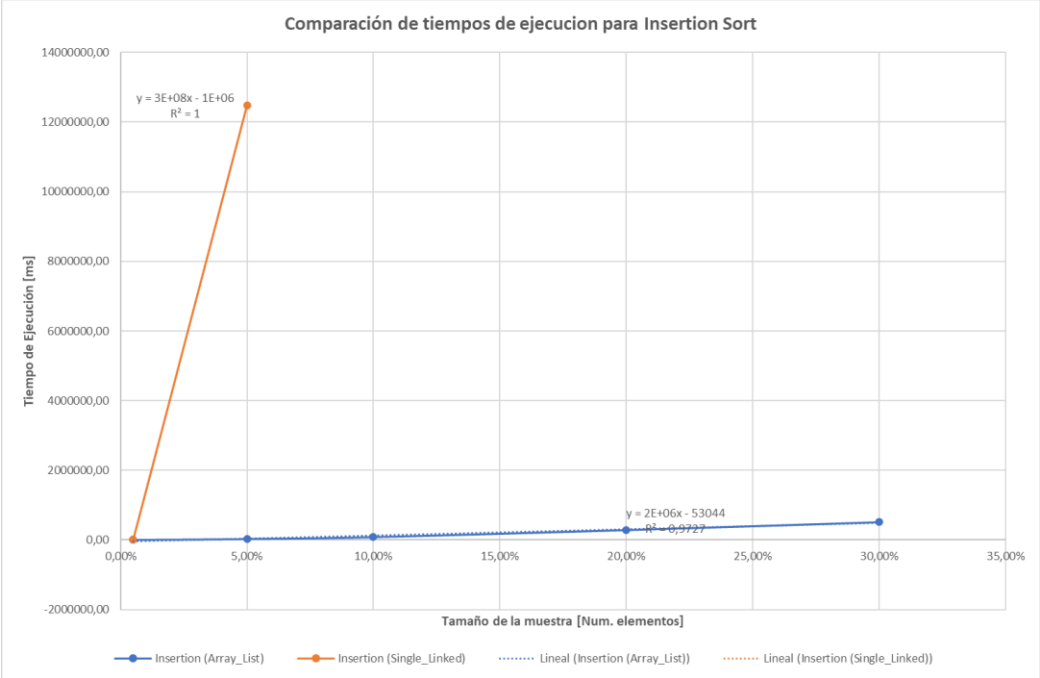
- Cinco gráficas generadas por los resultados de las pruebas de rendimiento en la **Maquina 2.**
  - Comparación de rendimiento ARRAYLIST.



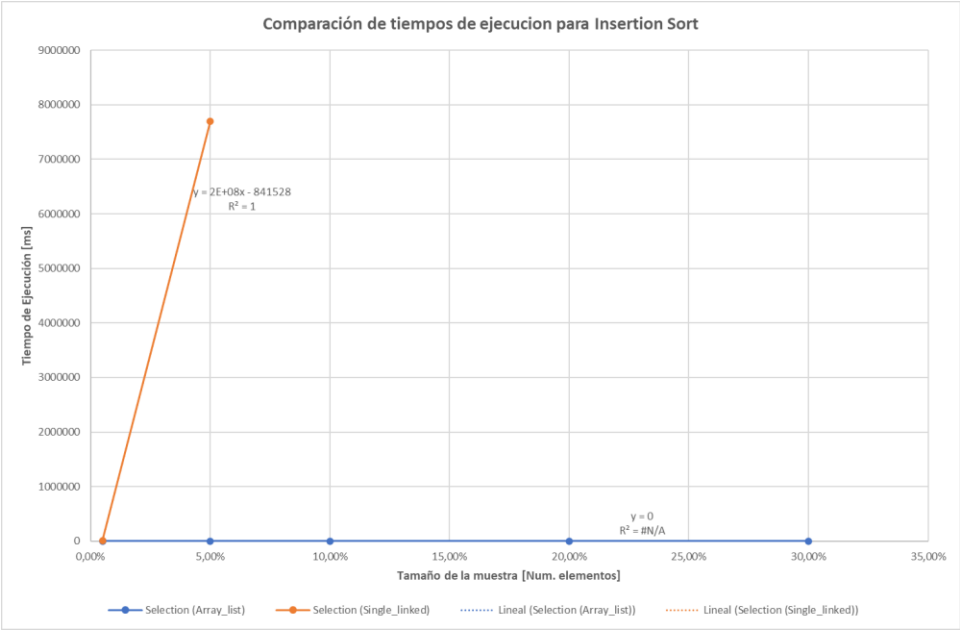
- Comparación de rendimiento LINKED_LIST.



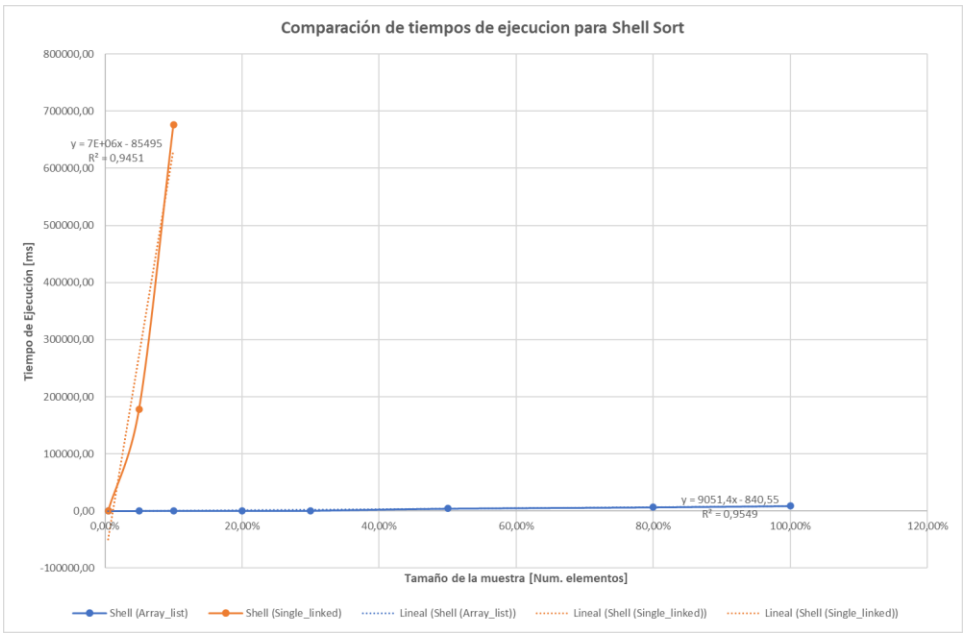
- Comparación de rendimiento para Insertion Sort.



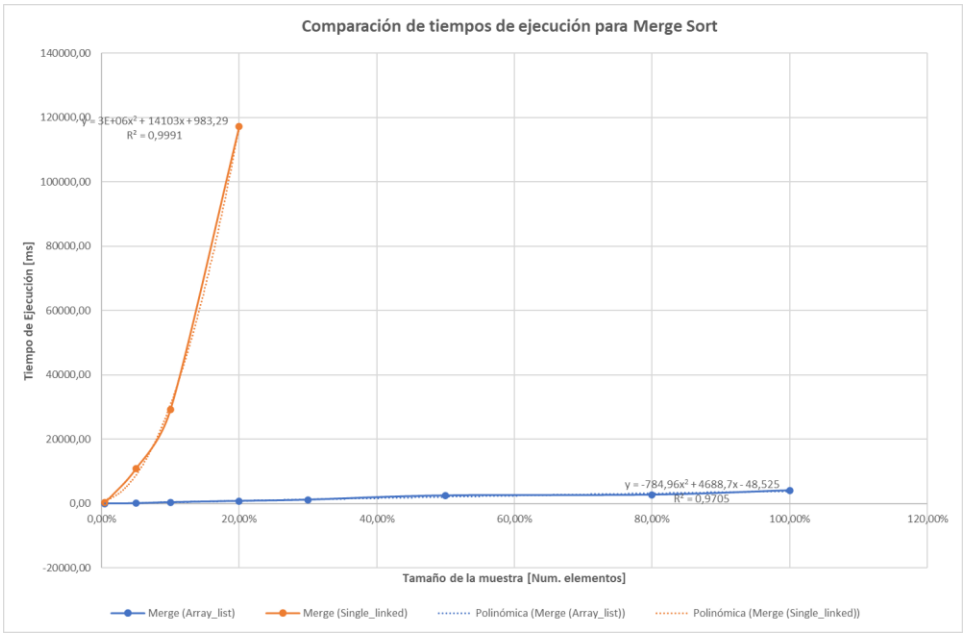
- Comparación de rendimiento para Selection Sort.



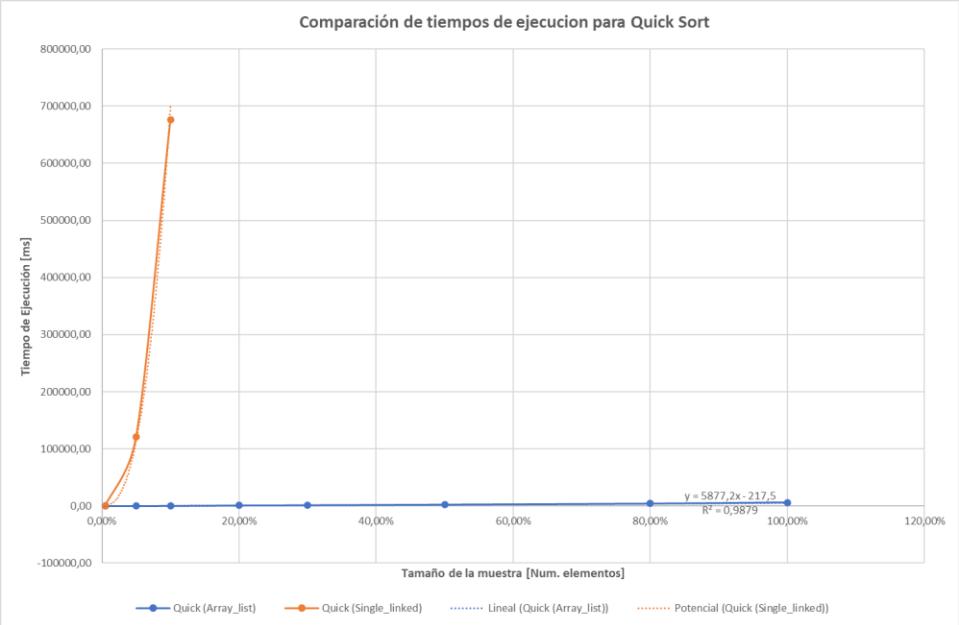
- Comparación de rendimiento para Shell Sort.



- Comparación de rendimiento para MergeSort.



- Comparación de rendimiento para QuickSort.



Preguntas de análisis

1) ¿El comportamiento de los algoritmos es acorde a lo enunciado teóricamente?
   - El comportamiento de los algoritmos en su mayoría va acorde con lo enunciado en la teoría, con esta base teórica podemos saber cómo se iban a comportar muy probablemente y en su totalidad se comportaron de la manera que se esperaba. El único caso donde la teoría no se cumple es con el ordenamiento Merge, que se espera que sea más lento que Quick, pero los valores registrados arrojaron otra conclusión.
2) ¿Existe alguna diferencia entre los resultados obtenidos al ejecutar las pruebas en diferentes máquinas?
   - Si, como podemos evidenciar en las tablas y en las gráficas podemos evidenciar que los datos que nos dieron llegan a ser bastantes diferentes en cuanto a números. Sin embargo, la teoría se sigue aplicando de la misma forma en ambas maquinas.
3) De existir diferencias, ¿A qué creen ustedes que se deben dichas diferencias?
   - Algunas diferencias que pueden llegar a hacer una diferencia entre las maquinas podría ser el procesador que estas tienen, las aplicaciones que están tanto abiertas como en segundo plano que estén gastando poder de procesamiento, y así mismo otros factores externos o del mismo sistema como por ejemplo que cuando no esté conectado con el fin de reducir el costo de energía sea más lento (en ciertos casos específicos, no es lo mismo en todos los computadores).
4) ¿Cuál Estructura de Datos es mejor utilizar si solo se tiene en cuenta los tiempos de ejecución de los algoritmos?
   - Usando los tiempos de ejecución de los algoritmos usados, podemos concluir que el mejor algoritmo para utilizar hasta el momento seria merge sort.
5) Para el caso analizado de ordenamiento de los artistas, teniendo en cuenta los resultados de tiempo reportados por todos los algoritmos de ordenamiento (iterativos y recursivos), proponga un ranking de los algoritmos de ordenamiento (de mayor eficiencia a menor (en tiempos de ejecución) para ordenar la mayor cantidad de artistas.

1. Merge sort
2. Quick sort
3. Shell sort
4. Selection sort
5. Insertion sort