OBSERVACIONES DE LA PRACTICA

Samuel Torres Cod 201632770

Alejandro Díaz-Granados Cod 202021008

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | Intel® Core(TM) i5-8250U CPU @ 1.60Hz 1.80Hz | 2.7 GHz Dual-Core Intel Core i5 |
| Memoria RAM (GB) | 4.0 GB | 8.0 GB |
| Sistema Operativo | Windows 10 64-bits | MacOS Catalina |

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

# **Maquina 1**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tamaño de la muestra (ARRAY\_LIST) | Insertion Sort (ms) | Selection Sort (ms) | Shell Sort (ms) | Merge Sort [ms] | Quick Sort [ms] |
| 1000 | 1052,08 | 1140,63 | 62,50 | 31,25 | 41,67 |
| 2000 | 4203,13 | 4942,71 | 114,59 | 67,71 | 78,13 |
| 4000 | 16739,59 | 20828,13 | 302,09 | 135,42 | 208,34 |
| 8000 | 67604,17 | 84244,80 | 708,34 | 270,84 | 385,42 |
| 16000 | 277546,88 | 344927,09 | 1942,71 | 828,13 | 661,46 |
| 32000 | 1140276,04 | 1396984,38 | 4083,34 | 1968,75 | 2041,67 |

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tamaño de la muestra (LINKED\_LIST) | Insertion Sort (ms) | Selection Sort (ms) | Shell Sort (ms) | Merge Sort [ms] | Quick Sort [ms] |
| 1000 | 68851,57 | 61429,69 | 2562,50 | 437,51 | 2617,19 |
| 2000 | 543015,63 | 513484,38 | 15539,07 | 1015,63 | 10960,94 |
| 4000 | 4390929,69 | 3934406,26 | 79695,32 | 5179,69 | 51812,51 |

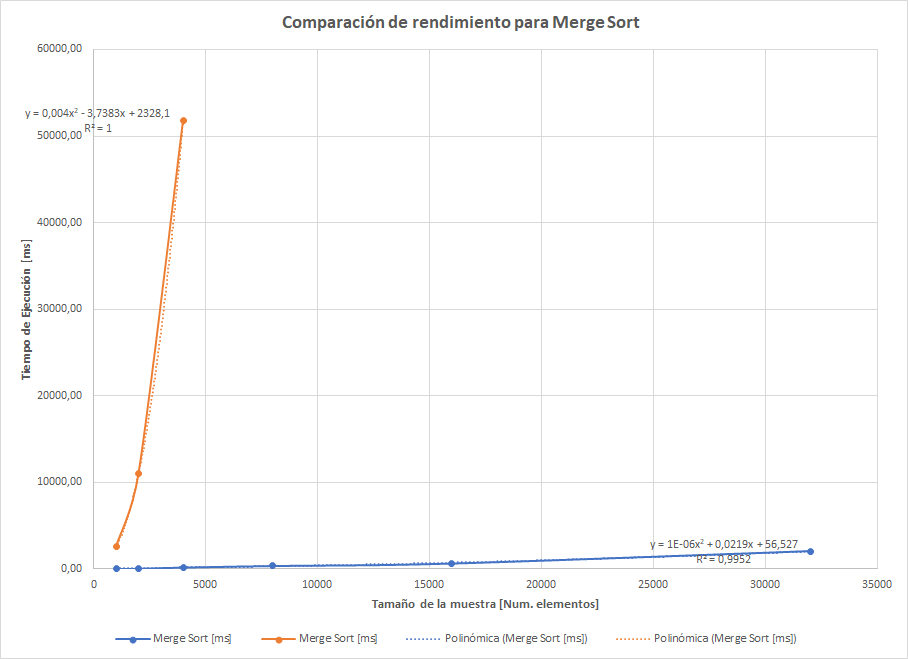
Tabla 3. Comparación de tiempos de ejecución para los ordenamientos iterativos en la representación lista enlazada.

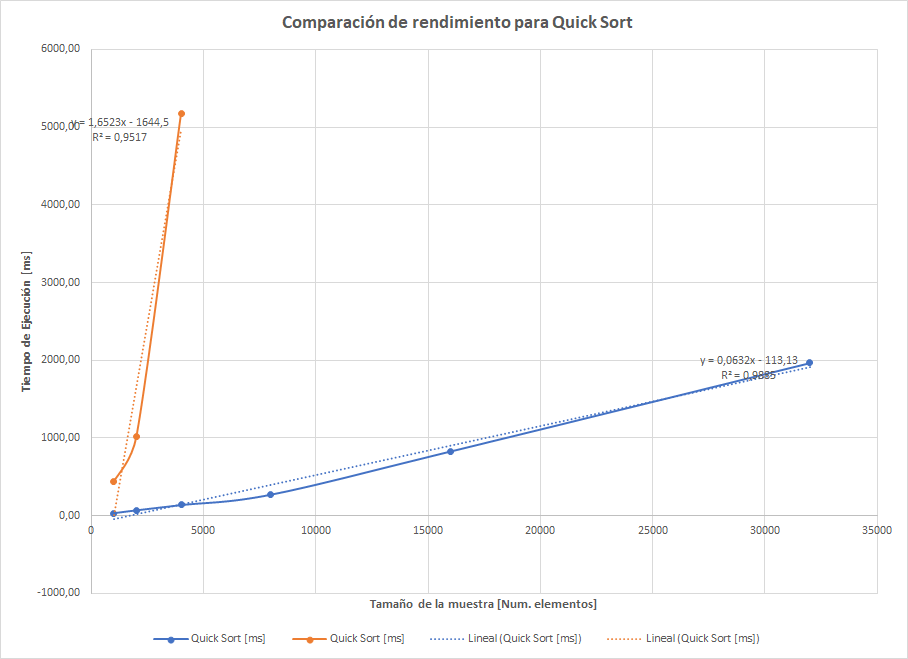
|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| Merge sort | X |  |
| Quicksort | X |  |

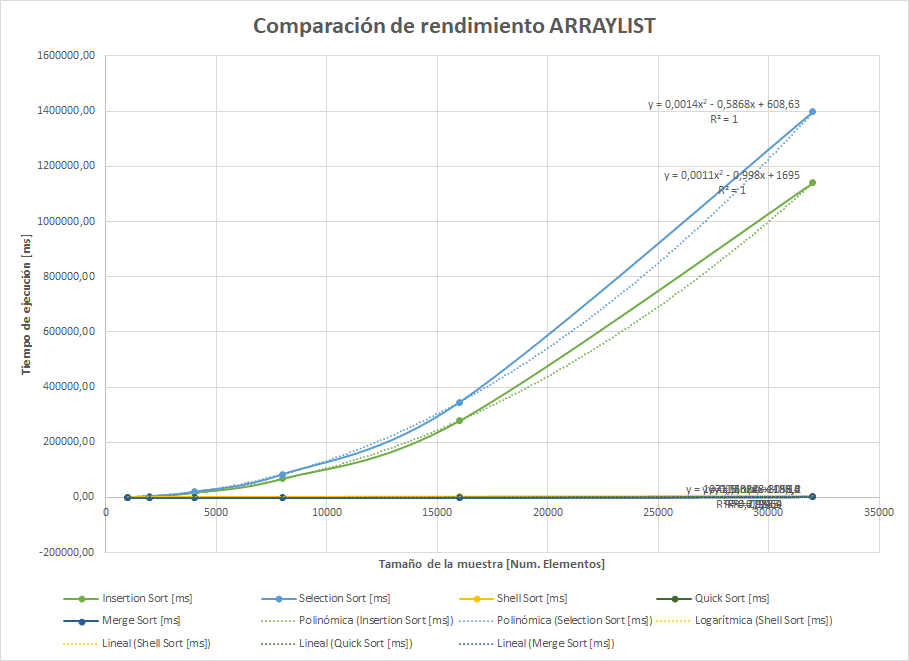
Tabla 4. Comparación de eficiencia de acuerdo con los algoritmos de ordenamientos y estructuras de datos utilizadas.

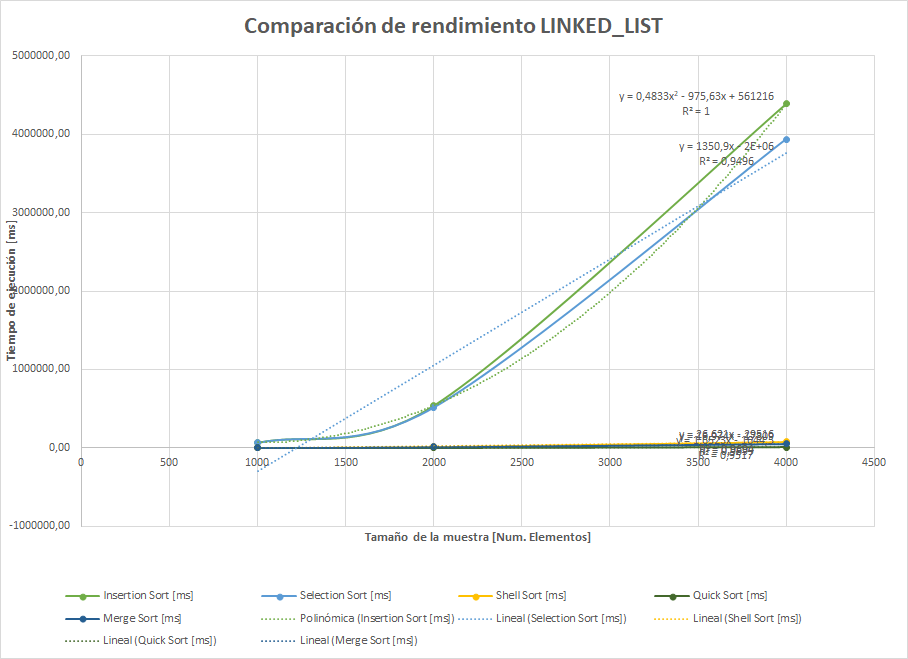
## **Graficas**

* Cinco gráficas generadas por los resultados de las pruebas de rendimiento en la **Maquina 1.**
  + Comparación de rendimiento para Quick Sort.
  + Comparación de rendimiento para Merge Sort
  + Comparación de rendimiento ARRAYLIST.
  + Comparación de rendimiento LINKED\_LIST.









# **Maquina 2**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tamaño de la muestra (ARRAY\_LIST) | Insertion Sort (ms) | Selection Sort (ms) | Shell Sort (ms) | Merge Sort [ms] | Quick Sort [ms] |
| 1000 | 730.882 | 1000.144 | 34.914 | 27.163 | 24.773 |
| 2000 | 2936.293 | 4214.768 | 82.389 | 59.438 | 57.569 |
| 4000 | 11813.200 | 15851.836 | 215.869 | 130.309 | 141.367 |
| 8000 | 48260.826 | 63484.043 | 517.862 | 288.489 | 321.626 |
| 16000 | 195815.397 | 230730.579 | 1194.402 | 615.728 | 609.211 |
| 32000 | 798763.134 | 1031518.835 | 2856.802 | 1339.300 | 1317.471 |

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tamaño de la muestra (LINKED\_LIST) | Insertion Sort (ms) | Selection Sort (ms) | Shell Sort (ms) | Merge Sort [ms] | Quick Sort [ms] |
| 1000 | 59528.167 | 49389.769 | 2653.668 | 274.104 | 2163.178 |
| 2000 | 479653.817 | 409698.016 | 13458.459 | 13458.459 | 9643.326 |
| 4000 | 3908324.789 | 3288351.012 | 70051.932 | 70051.932 | 44106.611 |

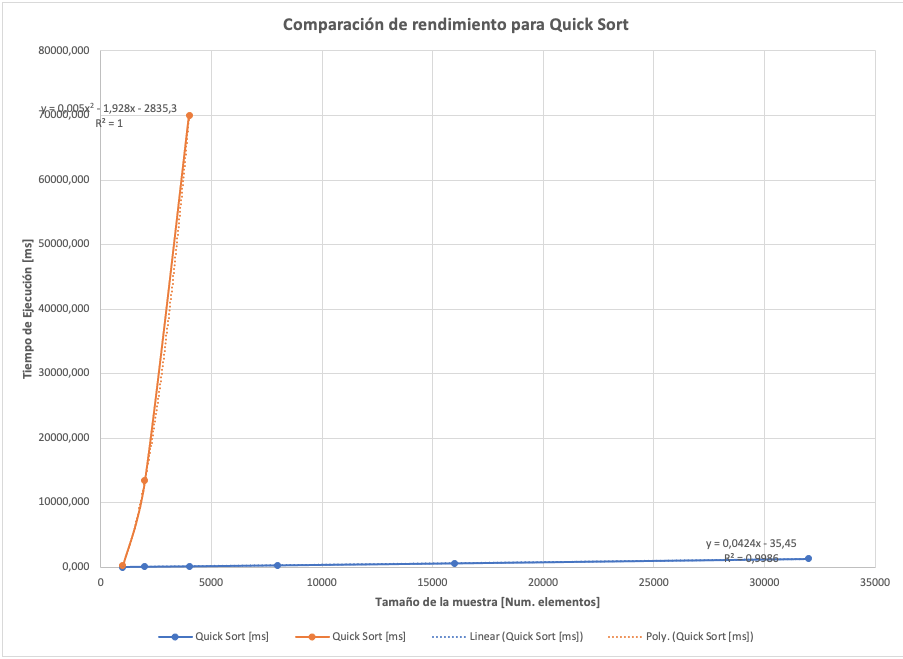
Tabla 3. Comparación de tiempos de ejecución para los ordenamientos iterativos en la representación lista enlazada.

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| Merge sort | X |  |
| Quick sort | X |  |

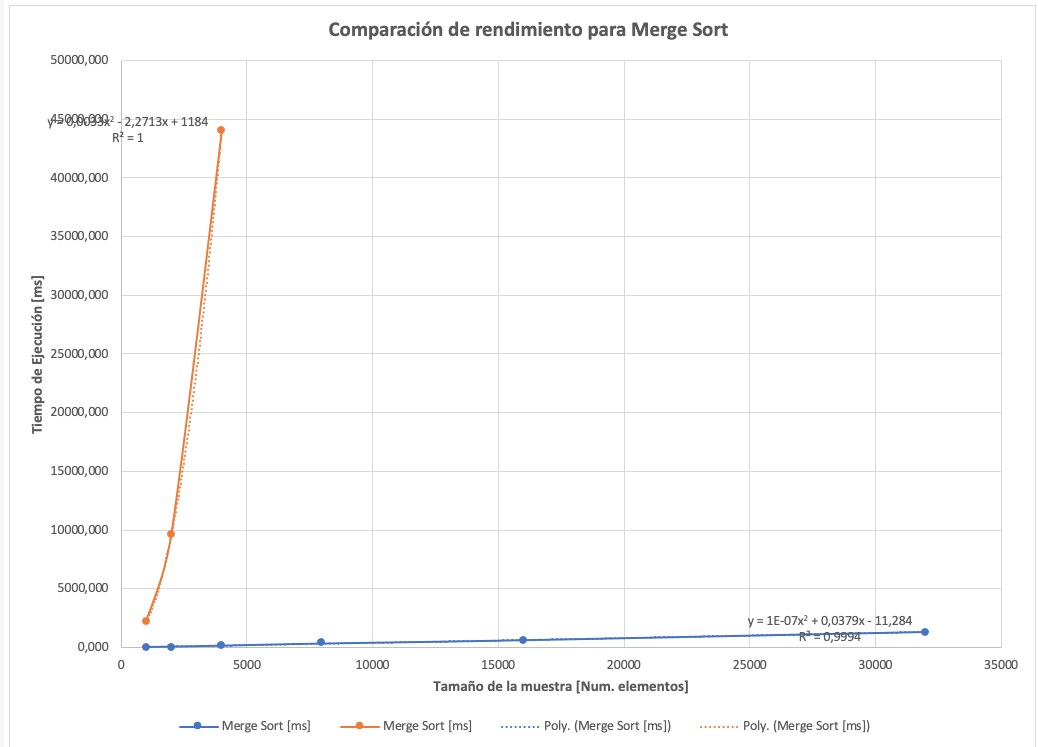
Tabla 4. Comparación de eficiencia de acuerdo con los algoritmos de ordenamientos y estructuras de datos utilizadas.

## **Graficas**

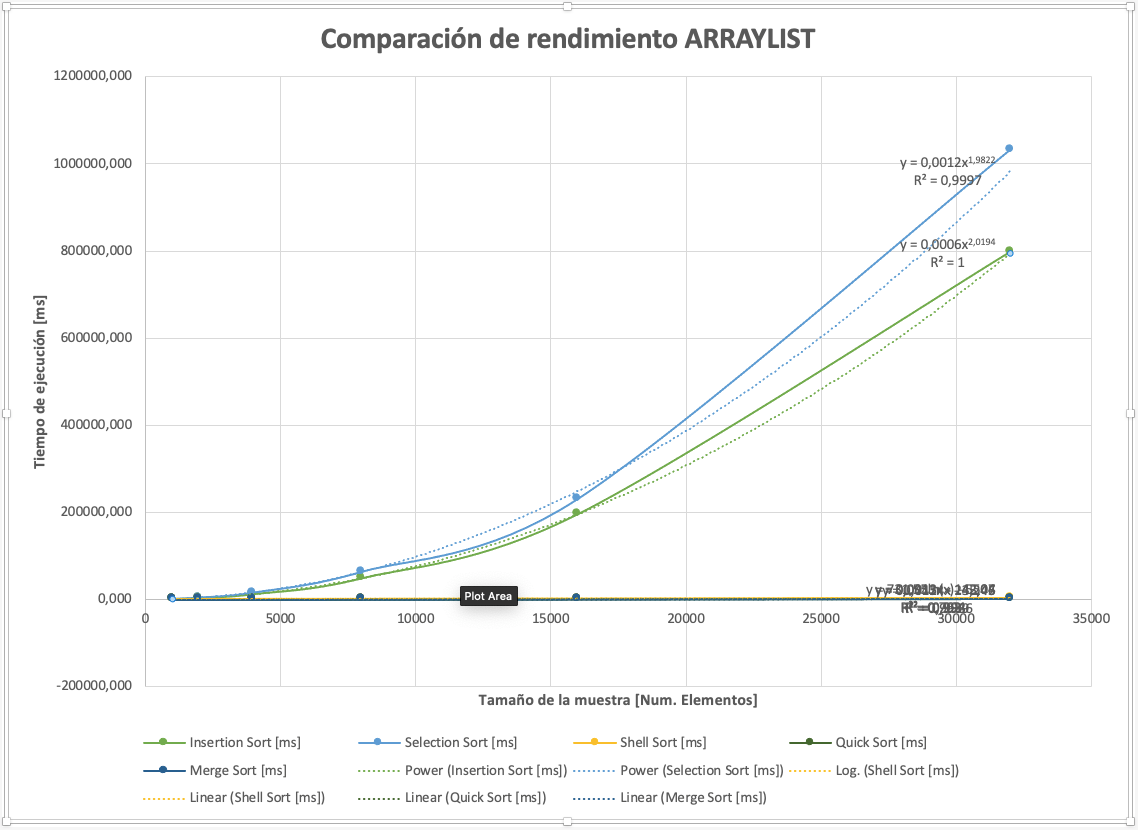
* Cinco gráficas generadas por los resultados de las pruebas de rendimiento en la **Maquina 2.**
  + Comparación de rendimiento para Quick Sort.



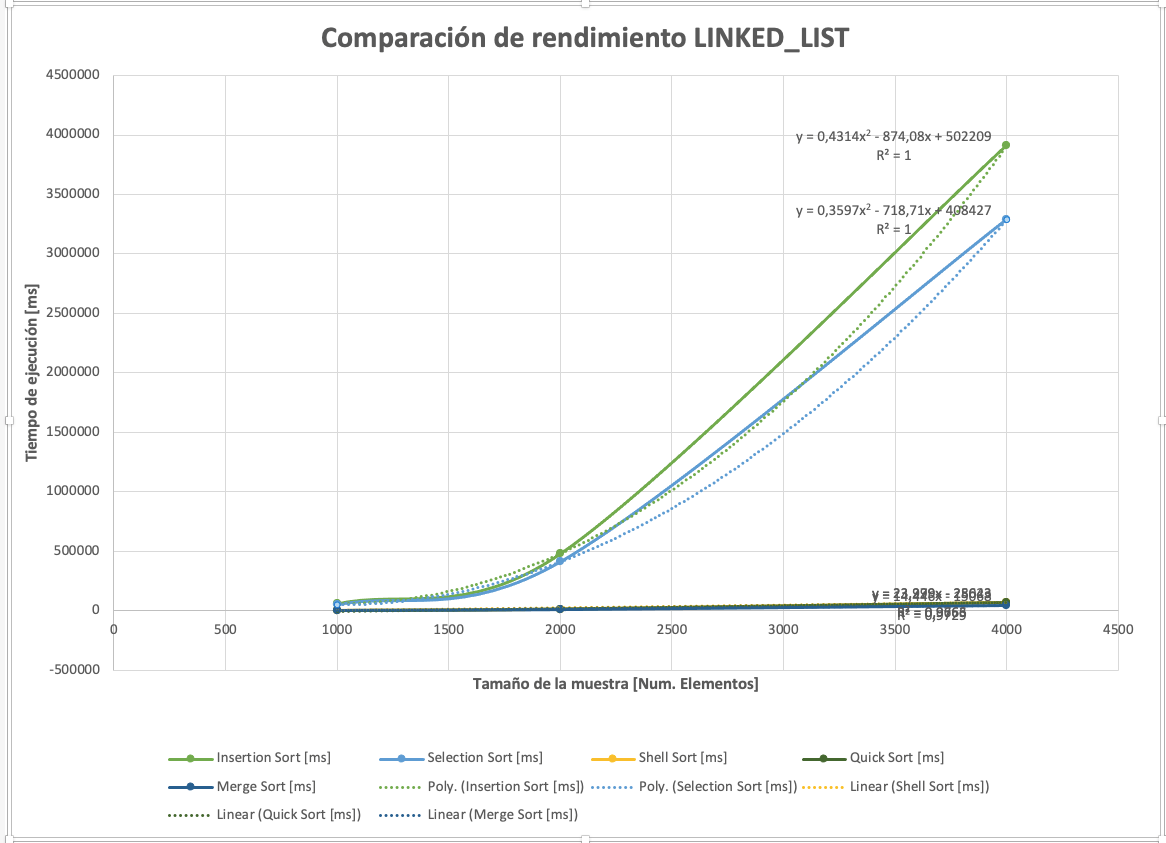
* + Comparación de rendimiento para Merge Sort



* + Comparación de rendimiento ARRAYLIST.



* + Comparación de rendimiento LINKED\_LIST.



# **Preguntas de análisis**

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

Sí, se comporta con lo esperado teoricamente. Se demuestra que a medida que incrementa la muestra, el algoritmo iterativo que supera a los demás es el QuickSort. Para muestras de datos pequeños, el algoritmo de Merge es mas eficiente.

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

Estas máquinas que tenían diferentes especificaciones (una mejor que la otra) tienen una diferencia visible en cuestión de resultados ya que todos los datos estuvieron en menos tiempo en la maquina 2

1. De existir diferencias, ¿A qué creen ustedes que se deben dichas diferencias?

El procesador y la memoria RAM de la maquina 2 son superiores a los de la maquina 1, esa parece ser la principal razón para que sea más rápido. Adicionalmente, el procesador de la máquina 2 es aproximadamente 1.5 veces más rápido.

1. ¿Cuál Estructura de Datos es mejor utilizar si solo se tiene en cuenta los tiempos de ejecución de los algoritmos?

Tomando de refereincia los ordenamientos Quick Sort y Merge Sort podemos apreciar que teniendo en cuenta las pruebas el ordenamiento Merge sort es un más rapido en los casos con muestras pequeñas, el Quick Sort es más eficienciente para muestras grandes.

Realice la comparación de los resultados obtenidos con los algoritmos iterativos analizados en el laboratorio anterior y respondan la siguiente pregunta:

Para el caso analizado de ordenamiento de los videos, teniendo en cuenta los resultados de tiempo reportados por todos los algoritmos de ordenamiento estudiados (iterativos y recursivos), proponga un ranking de los algoritmos de ordenamiento (de mayor eficiencia a menor eficiencia en tiempo) para ordenar la mayor cantidad de videos.

(ARRAY\_LIST)

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

(SINGLED\_LINKED)

1. Quick Sort
2. MergeSort
3. Shell Sort
4. Selection Sort
5. Insertion Sort