OBSERVACIONES DE LA PRACTICA

Juan Sebastián Hoyos - 201822167

Lyda Acuña - 201715885

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | 2.8 GHz Intel Core i7 cuatro nucleos | Intel® Celeron® N4000 CPU @ 1.10GHz 1.10GHz |
| Memoria RAM (GB) | 16.0 GB | 4.0 GB |
| Sistema Operativo | MacOs Catalina | Windows 10 64-bits |

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

# **Maquina 1**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (ARRAYLIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 816.39 | 1031.72 | 53.21 | 39.75 | 39.36 |
| 2000 | 3000.13 | 4281.37 | 126.54 | 83.73 | 81.68 |
| 4000 | 13191.82 | 16502.09 | 265.55 | 161.36 | 158.89 |
| 8000 | 45913.90 | 70964.43 | 599.87 | 377.10 | 344.64 |
| 16000 | 202301.14 | 302339.17 | 1450.07 | 690.01 | 683.13 |
| 32000 | 781997.87 | + 900000 | 3142.12 | 1443.90 | 1423.24 |
| 64000 | + 900000 | + 900000 | 7511.19 | 3564.47 | 3515.94 |
| 128000 | + 900000 | + 900000 | 19581.80 | 7677.48 | 7059.05 |
| 256000 | + 900000 | + 900000 | 44712.97 | 15293.71 | 13740.03 |
| 512000 | N/A | N/A | N/A | N/A | N/A |

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]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 51011,32 | 45293,22 | 2530,90 | 1789.60 | 228.09 |
| 2000 | 425419,13 | 369029,19 | 11790,37 | 9092.13 | 860.83 |
| 4000 | + 900000 | + 900000 | 55261,19 | 38594.86 | 4149.22 |
| 8000 | + 900000 | + 900000 | 281074,12 | 207452.54 | 14720.86 |
| 16000 | + 900000 | + 900000 | + 900000 | 735342.48 | 59177.56 |
| 32000 | + 900000 | + 900000 | + 900000 | +900000 | 296436.37 |
| 64000 | + 900000 | + 900000 | + 900000 | +900000 | +900000 |
| 128000 | + 900000 | + 900000 | + 900000 | +900000 | +900000 |
| 256000 | + 900000 | + 900000 | + 900000 | +900000 | +900000 |
| 512000 | N/A | N/A | N/A | N/A | N/A |

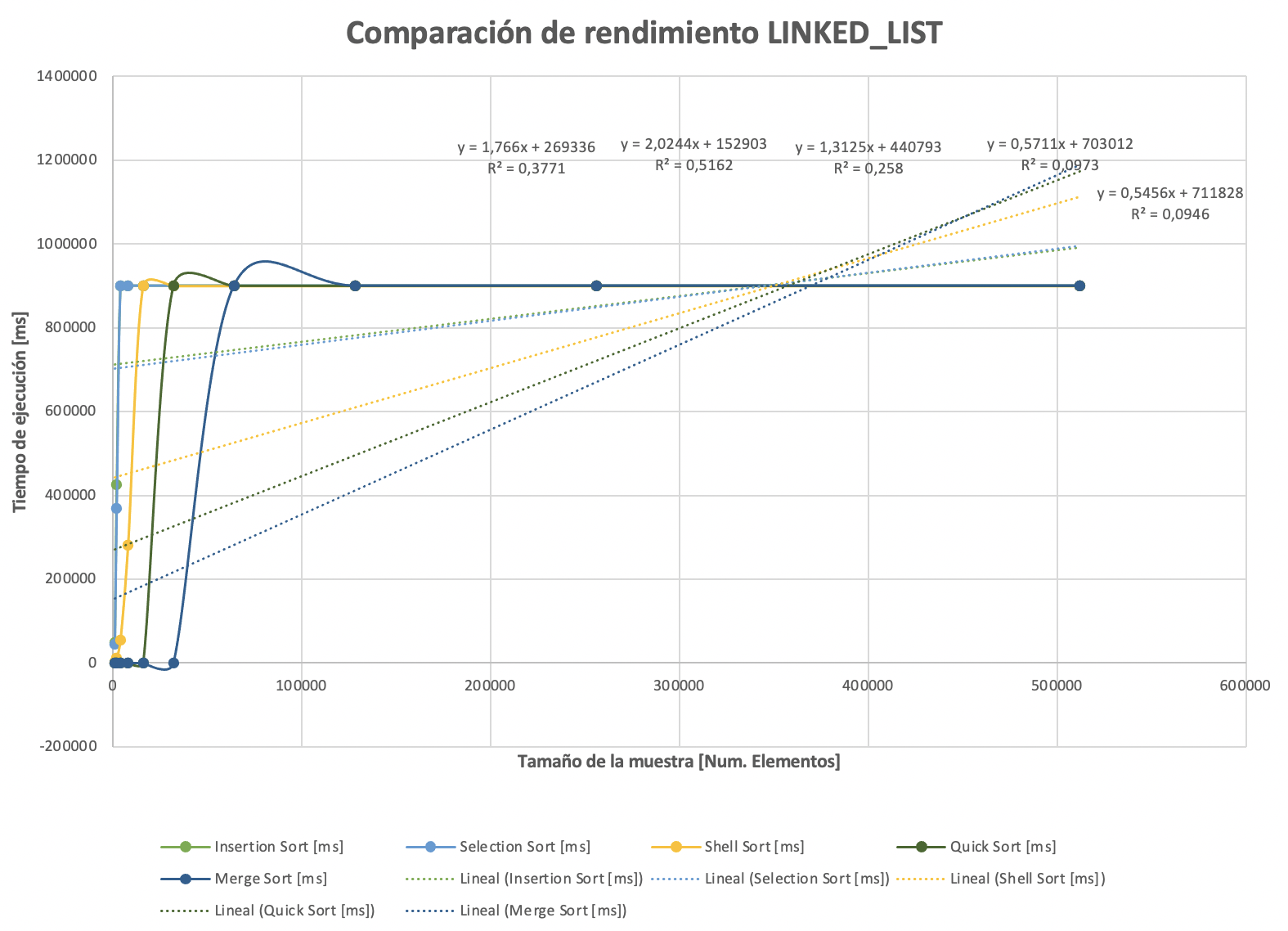
Tabla 3. Comparación de tiempos de ejecución para los ordenamientos iterativos en la representación lista enlazada.

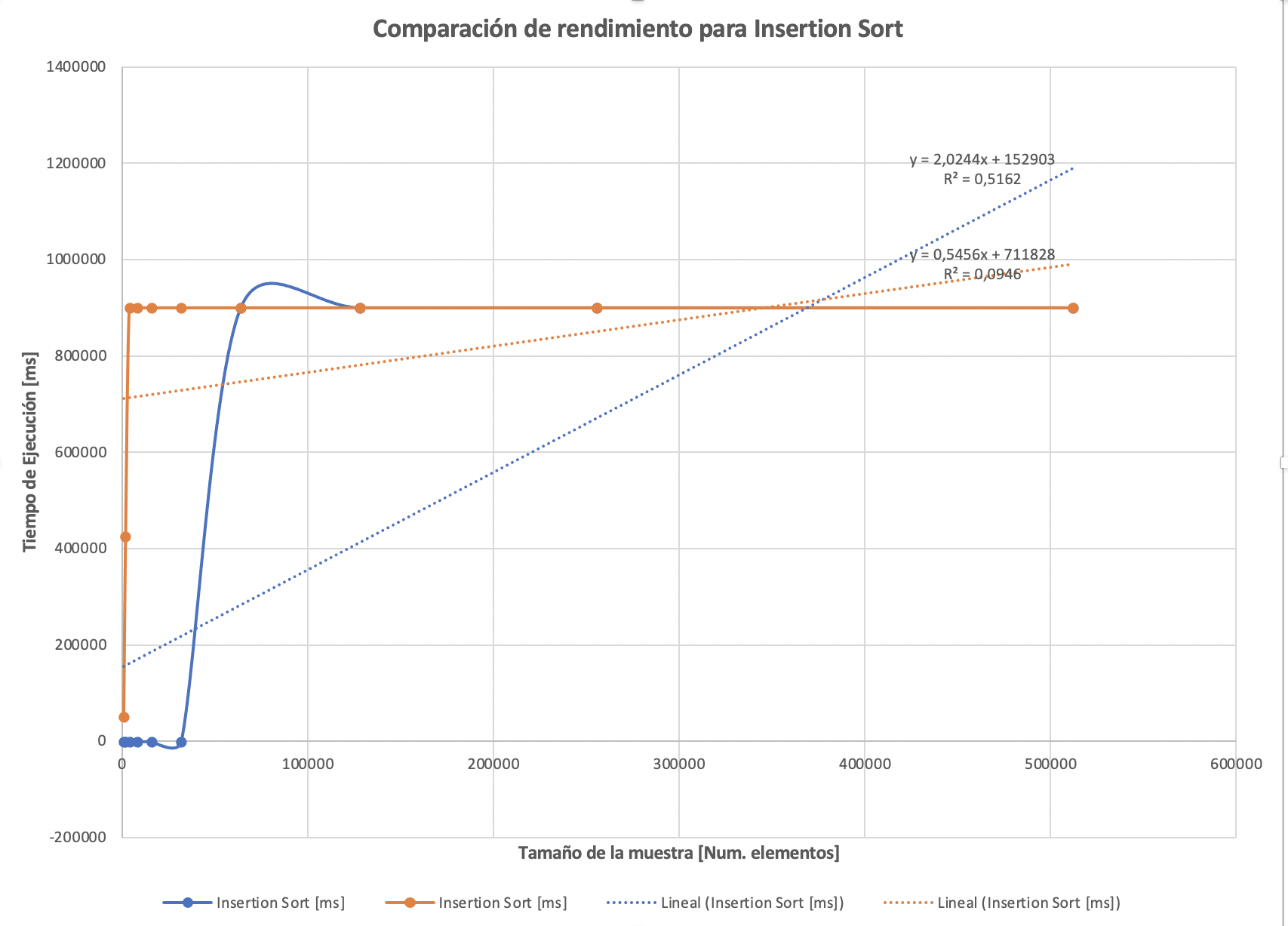
|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| Merge sort | **y = 1,5058x - 64044 R² = 0,7616** | **y = 2,0244x + 152903 R² = 0,5162** |
| Quick sort | **y = 1,5058x - 64044 R² = 0,7616** | **y = 1,766x + 269336 R² = 0,3771** |

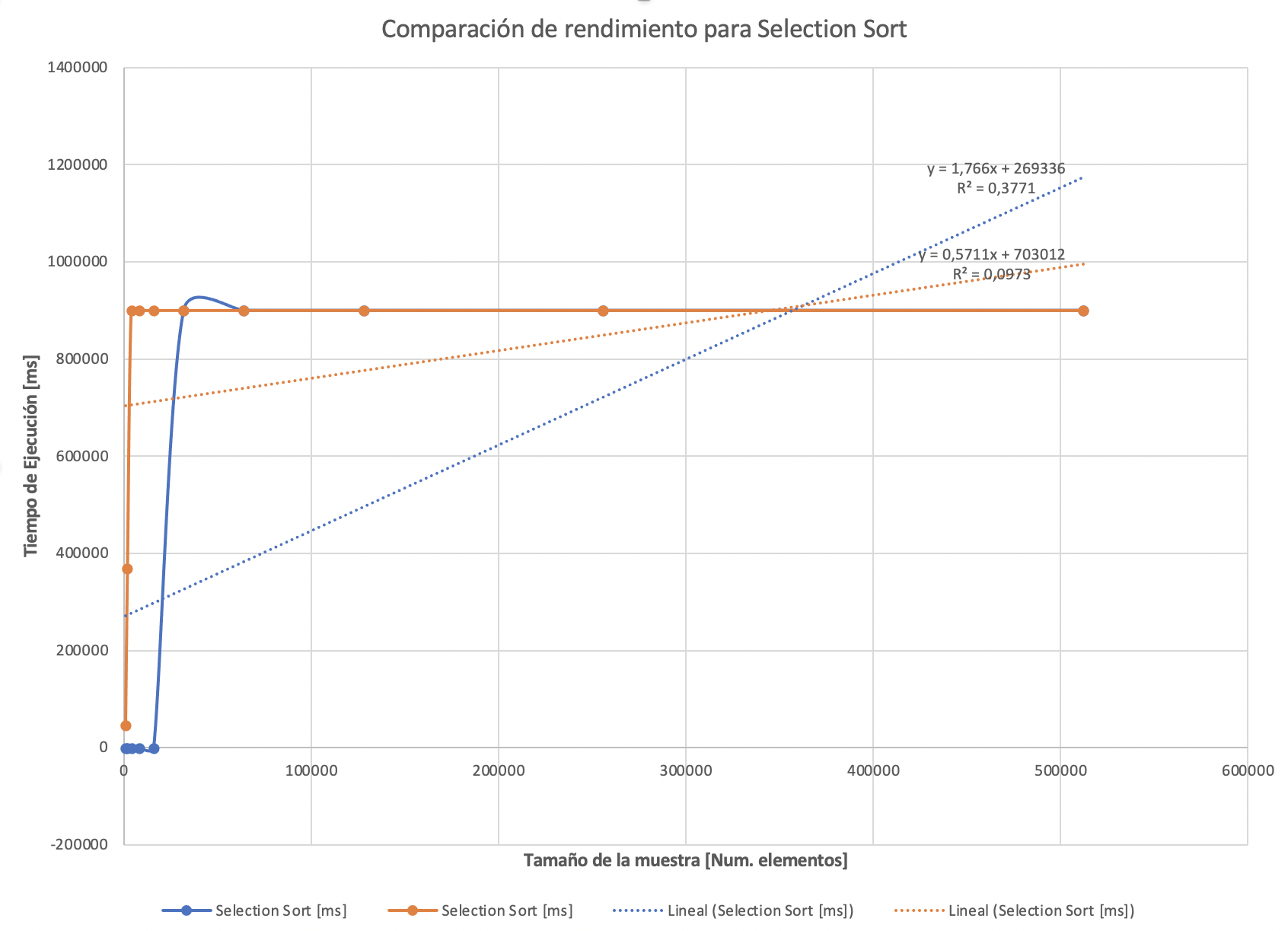
Tabla 4. Comparación de eficiencia de acuerdo con los algoritmos de ordenamientos y estructuras de datos utilizadas.

## **Graficas**

## 







# 

# 

# 

# **Maquina 2**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (ARRAYLIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 |  |  |  |  |  |
| 2000 |  |  |  |  |  |
| 4000 |  |  |  |  |  |
| 8000 |  |  |  |  |  |
| 16000 |  |  |  |  |  |
| 32000 |  |  |  |  |  |
| 64000 |  |  |  |  |  |
| 128000 |  |  |  |  |  |
| 256000 |  |  |  |  |  |
| 512000 |  |  |  |  |  |

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]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 |  |  |  |  |  |
| 2000 |  |  |  |  |  |
| 4000 |  |  |  |  |  |
| 8000 |  |  |  |  |  |
| 16000 |  |  |  |  |  |
| 32000 |  |  |  |  |  |
| 64000 |  |  |  |  |  |
| 128000 |  |  |  |  |  |
| 256000 |  |  |  |  |  |
| 512000 |  |  |  |  |  |

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

Efectivamente si existen diferencias entre los resultados obtenidos en ambas máquinas.

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

Creemos que estas diferencias corresponden a factores como el sistema operativo, el espacio de memoria y el procesador de cada dispositivo, ya que obtuvimos resultados con mayor rapidez en la máquina “más rapida” y con características superiores a la otra máquina en cuanto a los 3 puntos mencionados.

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

Si solo se tiene en cuenta los tiempos de ejecución, consideramos que el mejor algoritmo para ser empleado es Merge Sort, pues como se puede evidenciar en las tablas presentadas previamente, este algoritmo es el que presenta los menores tiempos de ejecución.

1. **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.**

De acuerdo a nuestros resultados, este es el ranking que planteamos ordenado de mayor eficiencia (Merge Sort) a menor eficiencia (Selection Sort).

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