OBSERVACIONES DE LA PRACTICA

Estudiante 1 Cod 2020213371

Estudiante 2 Cod 202022217

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | I5-9300H CPU 2.40GHZ | AMD Ryzen 5 3500u  2.10GHz |
| Memoria RAM (GB) | 16 | 12 |
| Sistema Operativo | Windows 10 | Windows 10 |

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 | 843.75 | 1119.70 | 67.70 | 20.8 | 36.4 |
| 2000 | 3294.62 | 3760.41 | 130.20 | 71.3 | 78.1 |
| 4000 | 16171.83 | 16057.27 | 203.12 | 171.83 | 182.24 |
| 8000 | 57557.28 | 68317.63 | 608.37 | 364.63 | 333.3 |
| 16000 | 236364.33 | 266760.40 | 1260.40 | 619.76 | 671.87 |
| 32000 | Tiempo Excedido | Tiempo Excedido | 3817.70 | 1354.16 | 1437.46 |
| 64000 |  |  | 9083.33 | 3244.76 | 2754.97 |
| 128000 |  |  | 16411.4 | 6453.12 | 6635.36 |
| 256000 |  |  | 44020.8 | 14911.45 | 14734.33 |
| 375942 |  |  |  | 31391.13 | 22312.33 |

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

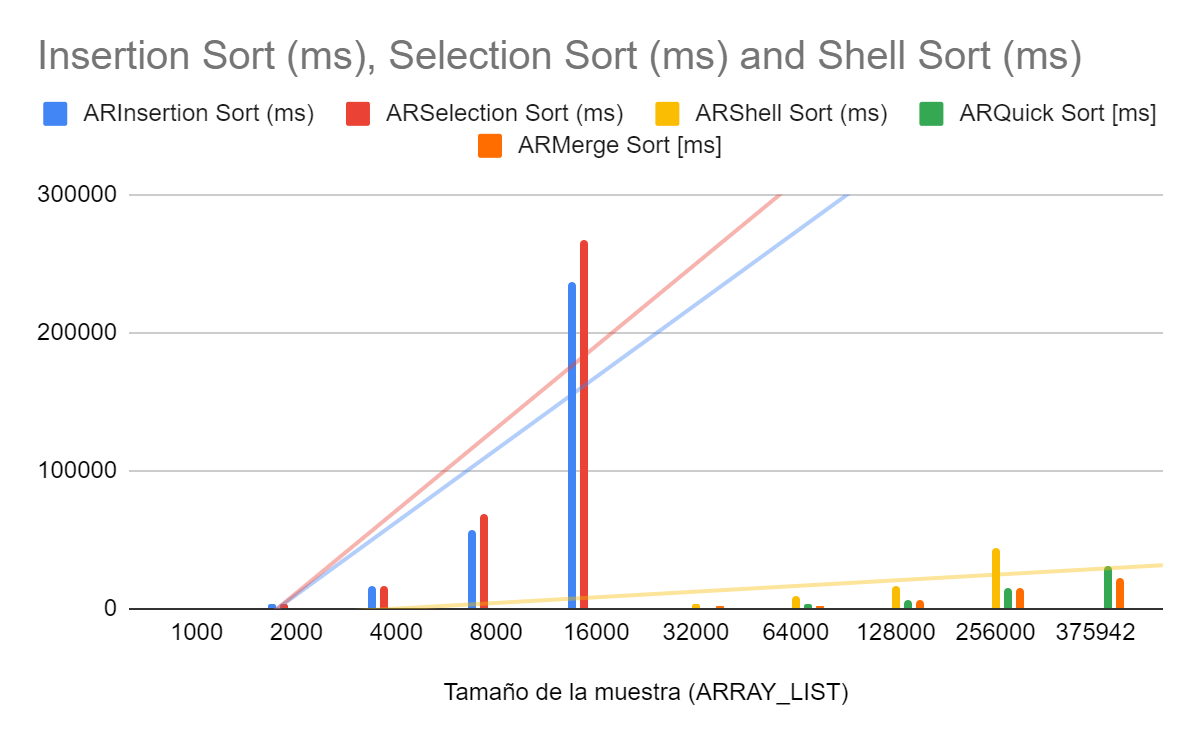
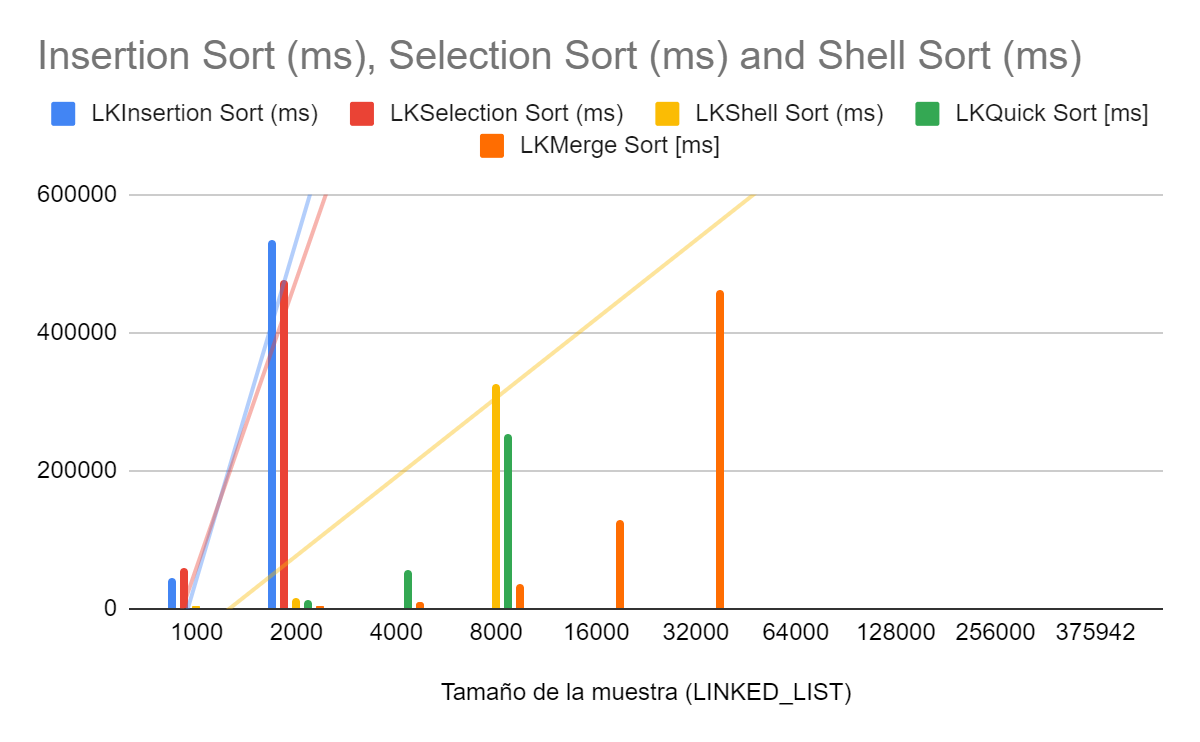
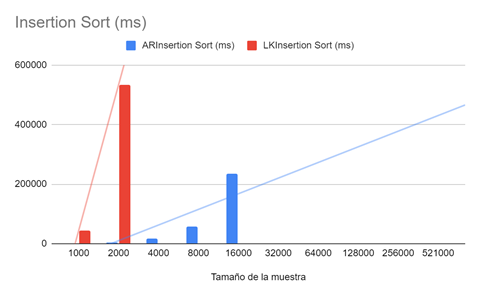
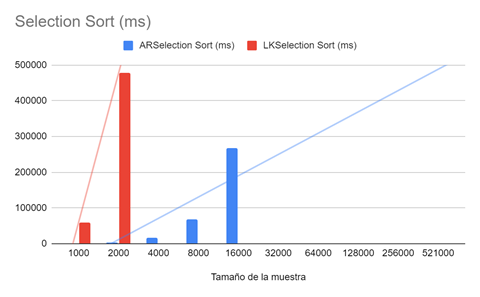
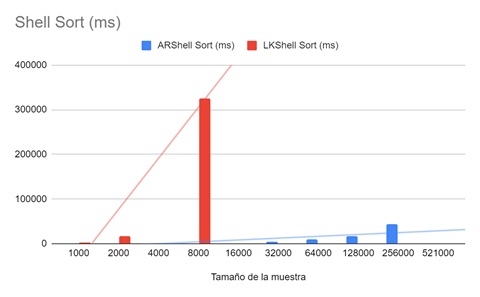
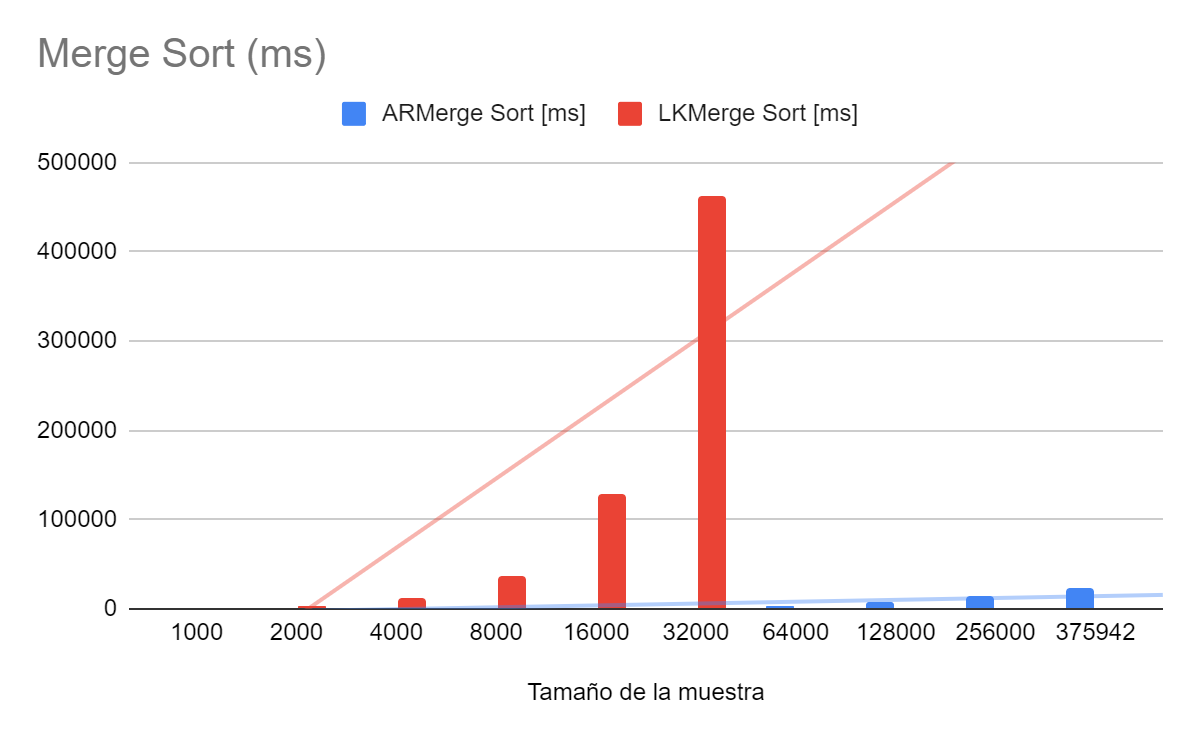
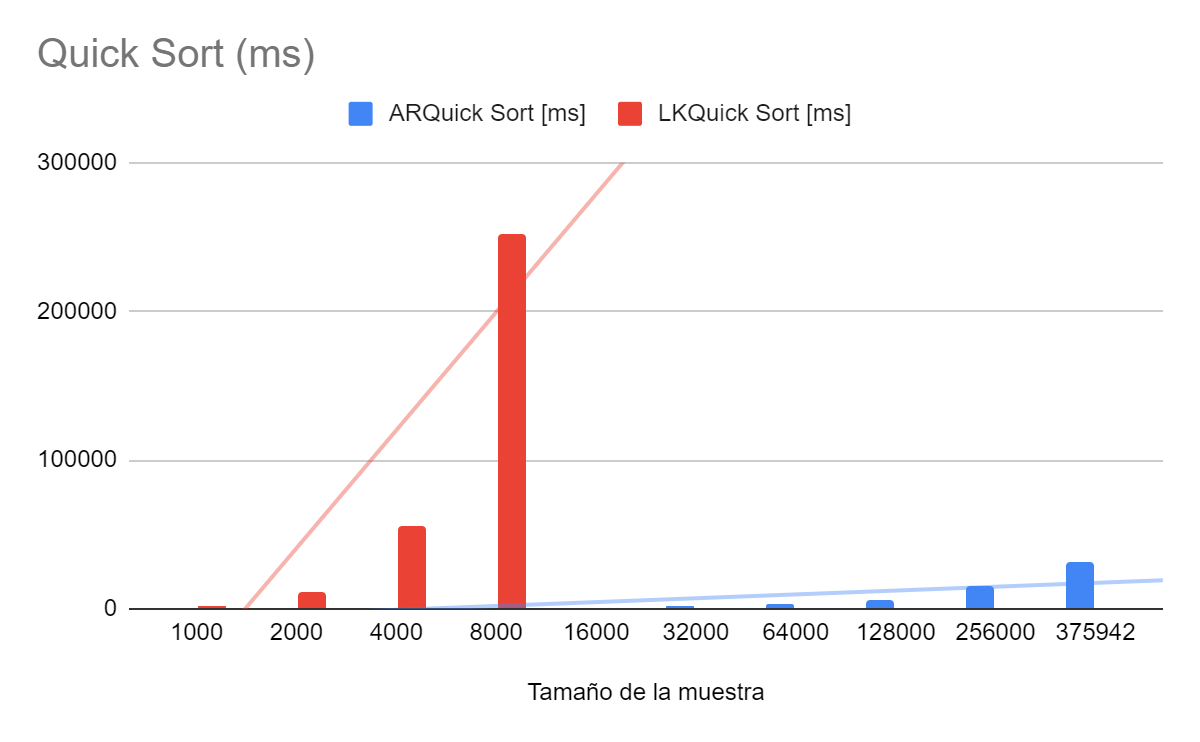
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (LINKED\_LIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 43586.45 | 60241.00 | 3291.66 | 2437.5 | 697.7 |
| 2000 | 533874.99 | 477401.04 | 16255.2 | 11845.62 | 3515.62 |
| 4000 | Tiempo Excedido | Tiempo Excedido | 65258.7 | 55187.4 | 11406.25 |
| 8000 |  |  | 325853 | 252406.25 | 36203.12 |
| 16000 |  |  | Tiempo Excedido | Tiempo Excedido | 129296.87 |
| 32000 |  |  |  |  | 462796.87 |
| 64000 |  |  |  |  | Tiempo excedido |
| 128000 |  |  |  |  |  |
| 256000 |  |  |  |  |  |
| 375942 |  |  |  |  |  |

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 | Eficiencia ligeramente menor a Quicksort hasta llegar a 32000 datos,de ahí para adelante Merge es mas eficiente | Mucha mejor eficiencia en comparación con quicksort pero no es muy eficiente aún así. |
| Quick sort | Mas eficiente que Mergesort hasta los 32000 datos.De ahí para adelante decae en comparación con merge | Una eficiencia muy baja comparado con MergeSort,a pesar de que Mergesort también sea poco eficiente |

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**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (ARRAYLIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 80046.88 | 69593.75 | 3843.75 | 78.125 | 62.5 |
| 2000 | 643796.88 | 813500.00 | 18218.60 | 125 | 140.625 |
| 4000 | Tiempo Excedido | Tiempo Excedido | 119437.5 | 296.875 | 281.25 |
| 8000 |  |  | 587218.7 | 734.375 | 671.875 |
| 16000 |  |  | Tiempo Excedido | 1437.5 | 1515.625 |
| 32000 |  |  |  | 3000.0 | 3125.0 |
| 64000 |  |  |  | 6328.125 | 6890.625 |
| 128000 |  |  |  | 13484.37 | 13437.5 |
| 256000 |  |  |  | 31718.75 | 29109.37 |
| 512000 | Excede tamaño | Excede tamaño | Excede tamaño | Excede tamaño | Excede tamaño |

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

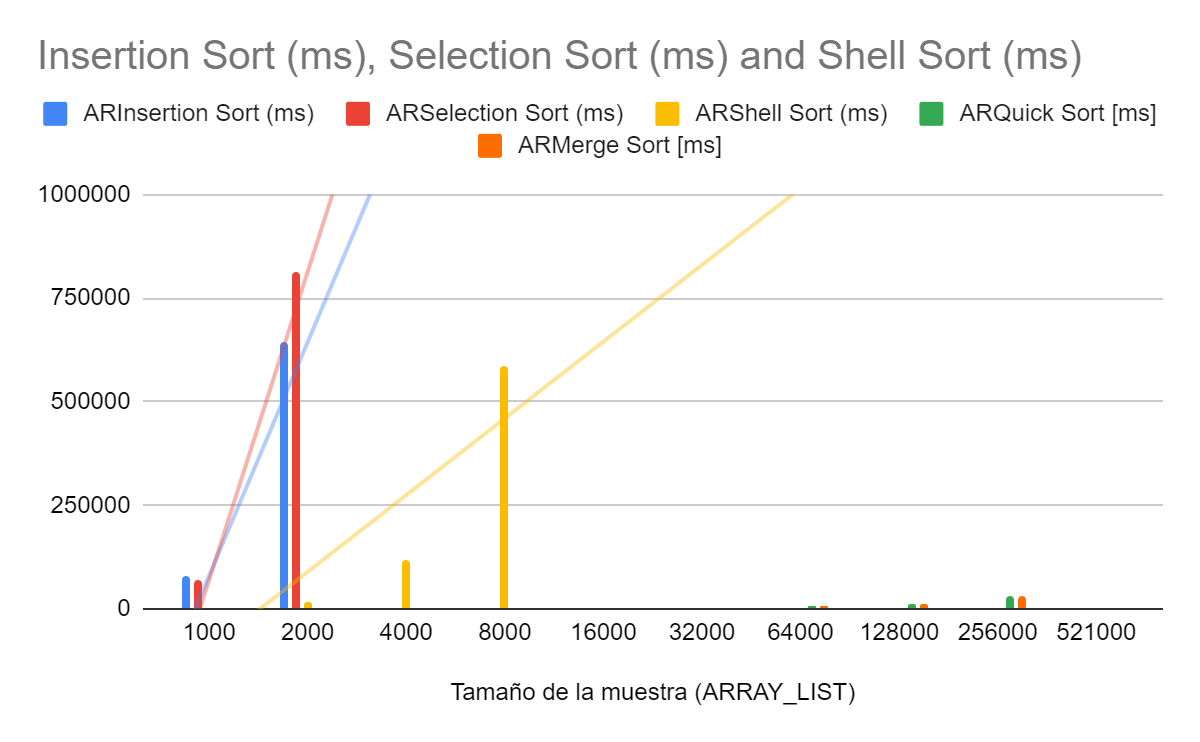
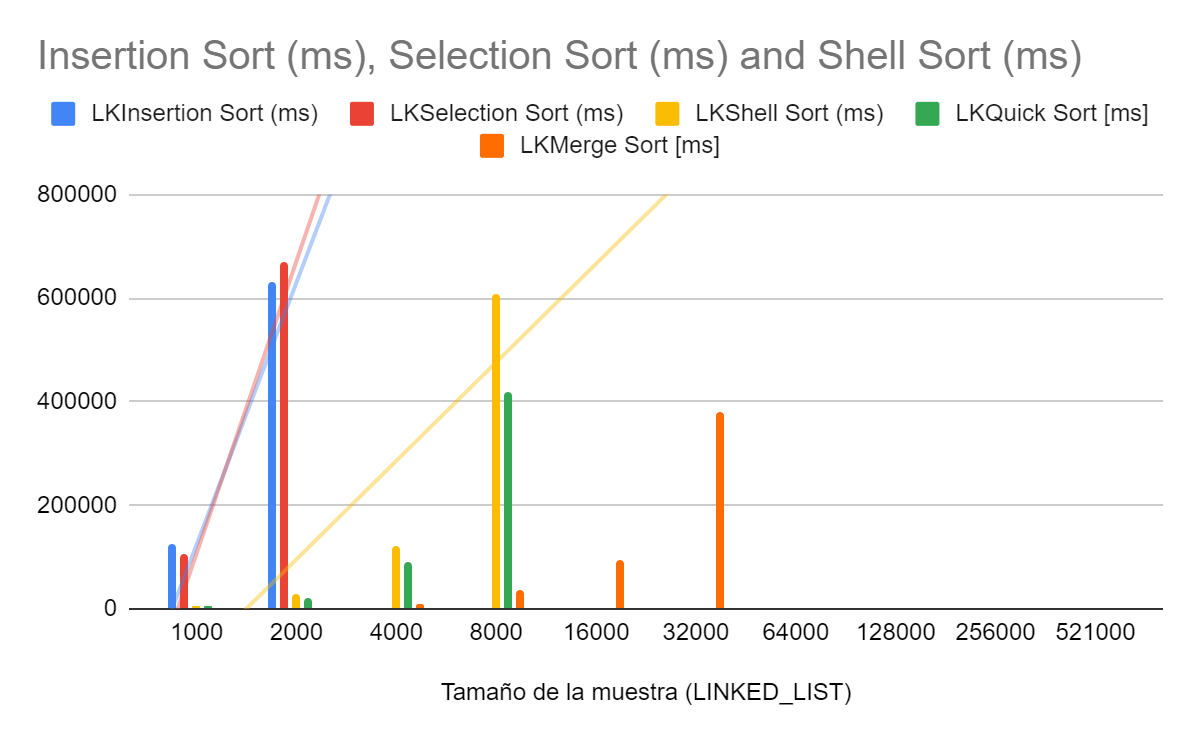
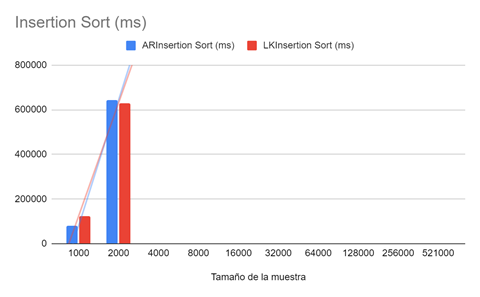
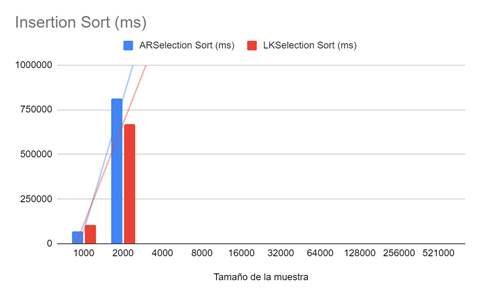
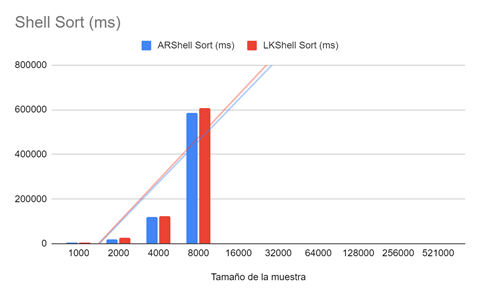
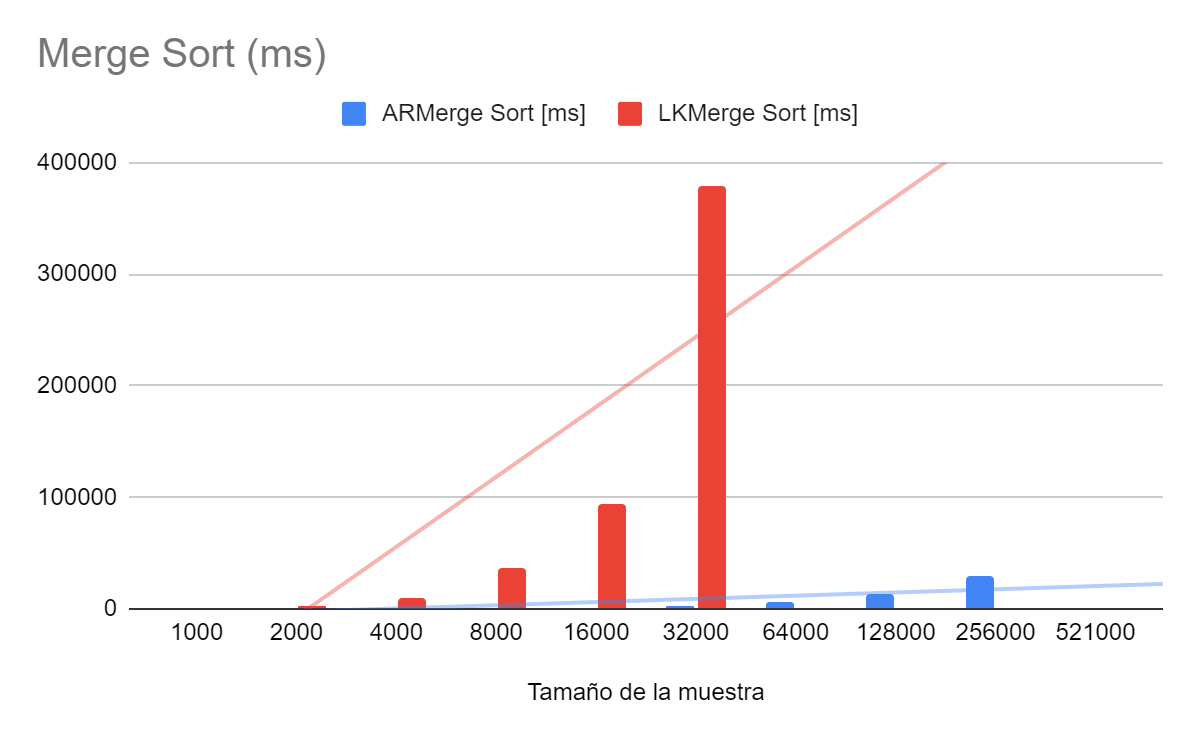
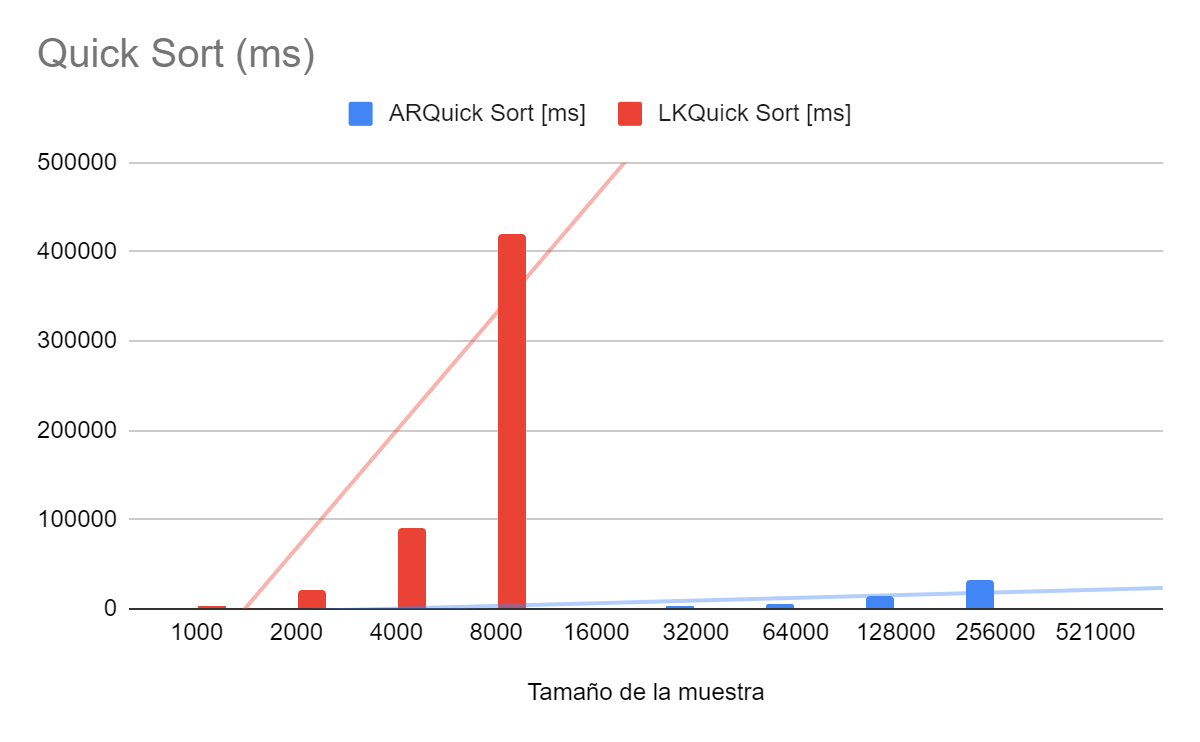
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (LINKED\_LIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 123984.375 | 107187.50 | 6031.25 | 4312.5 | 609.375 |
| 2000 | 630437.50 | 671703.125 | 27171.87 | 20765.62 | 2421.875 |
| 4000 | Excede tiempo | Excede tiempo | 121812.5 | 90640.62 | 9656.25 |
| 8000 |  |  | 608593.7 | 419546.8 | 35843.75 |
| 16000 |  |  | Excede tiempo | Excede tiempo | 94328.12 |
| 32000 |  |  |  |  | 379171.8 |
| 64000 |  |  |  |  | Excede tiempo |
| 128000 |  |  |  |  |  |
| 256000 |  |  |  |  |  |
| 512000 | Excede tamaño | Excede tamaño | Excede tamaño | Excede tamaño | Excede tamaño |

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 | Algoritmo de mejor eficiencia en general, aunque por la mayor parte permaneció prácticamente igualado a quick sort sin mostrar mayor tendencia de diferencia. | Al utilizar linked, la eficiente se vuelve inmediatamente inferior en comparación a array, pero merge se vuelve más eficiente que quick sort desde el comienzo por una buena cantidad. |
| Quick sort | Muy similar a merge sort y hasta superior en unos cuantos conjuntos de datos marcados. Bastante eficiente. | A pesar de su mayor eficiencia en comparación con los algoritmos anteriores, el arlgoritmo linked y quick es el más ineficiente entre los mostrados en esta tabla. |

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?

* Concierne a los algoritmos nuevos, se cumple apropiadamente la teoría de que quick sort es más eficiente cuando se trata de muestras de datos pequeñas, mientras que merge es más eficiente entre más aumente de tamaño.

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

* En la máquina 1 la rapidez de las operaciones fue mayor en todos los casos, y se pudo evidenciar con más claridad la diferencia en efectividad entre quick y merge.

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

* Como se enunció en el laboratorio anterior, los principales factores que pudieron causar estas diferencias incluyen la diferencia en velocidad de procesamiento de los computadores (2.40 contra 2.10GHz) o los procesos de segundo plano que se pudieron desapercibidamente estar desarrollando.

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

* Teniendo en cuenta únicamente los tiempos, la estructura superior para datos pequeños es la quick de arreglo, mientras que para datos grandes es merge de arreglo.

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.

* Si se trata de ordenar la mayor cantidad de videos, basándose en los datos, el ranking de algoritmos sería: 1. Merge-Array, 2. Quick-Array, 3. Shell-Array, 4. Merge-Linked, 5. Insertion-Array, 6. Selection-Array, 7. Quick-Linked, 8. Shell-Linked, 9. Insertion-Linked, 10. Selection-Linked