David Santiago Vargas Prada 202013826

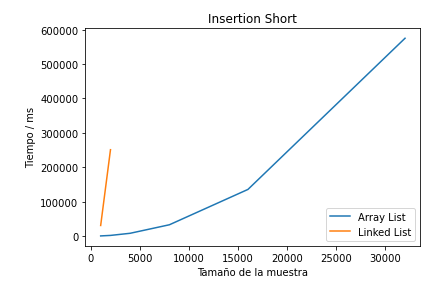
Juan Esteban Cuellar Argotty 202014258

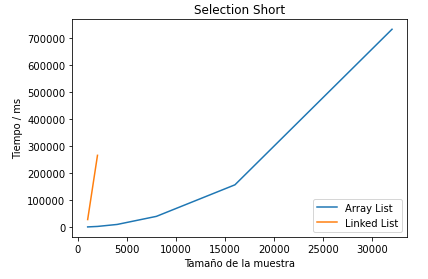
|  |  |  |
| --- | --- | --- |
|  | *Máquina 1* | *Máquina 2* |
| **Procesadores** | AMD Ryzen 5 3600 6-Core Processor - 3.60 GHz | Intel ® Core(™)i7-9750H CPU@2.60GHz 2.59GHz |
| **Memoria RAM(GB)** | 16 GB | 8 GB |
| **Sistema Operativo** | Windows 10 Pro-64 bits | Windows 10 Pro-64 bits |

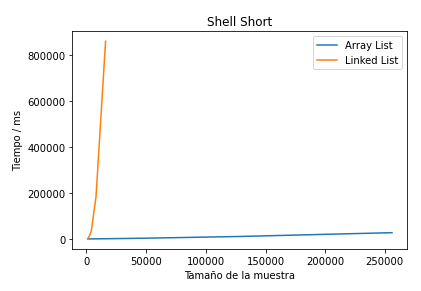
**Datos y graficas Máquina 1**

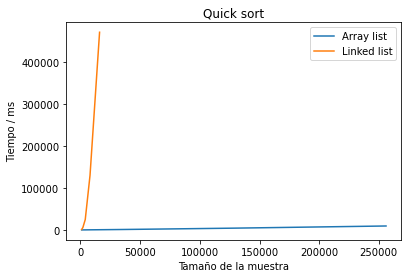
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (Array\_List)** | **Insertion Sort (ms)** | **Selection Sort (ms)** | **Shell Sort (ms)** | **Quick Sort(ms)** | **Merge Sort(ms)** |
| 1000 | 515.62 | 609.37 | 31.25 | 15.625 | 15.625 |
| 2000 | 1984.37 | 2406.25 | 62.5 | 46.875 | 46.875 |
| 4000 | 8156.25 | 9734.37 | 156.25 | 93.75 | 93.75 |
| 8000 | 32921.87 | 39656.25 | 359.375 | 250.0 | 203.125 |
| 16000 | 135640.62 | 156437.5 | 781.25 | 468.75 | 406.25 |
| 32000 | 575093.75 | 732265.62 | 1890.62 | 921.875 | 906.25 |
| 64000 | Tiempo excedido | Tiempo excedido | 4515.62 | 1984.375 | 1953.125 |
| 128000 | Tiempo excedido | Tiempo excedido | 10609.37 | 4203.125 | 4125.0 |
| 256000 | Tiempo excedido | Tiempo excedido | 27187.5 | 9375.0 | 8828.125 |
| 512000 | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos |

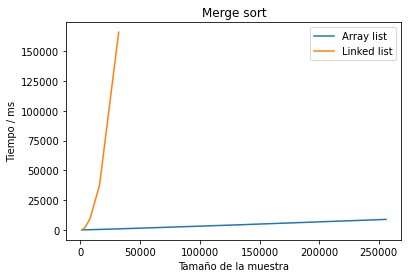
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (Linked\_List)** | **Insertion Sort (ms)** | **Selection Sort (ms)** | **Shell Sort (ms)** | **Quick Sort(ms)** | **Merge Sort(ms)** |
| 1000 | 30859.37 | 27843.75 | 1531.25 | 1140.625 | 156.25 |
| 2000 | 251125.0 | 265593.75 | 8078.12 | 5640.625 | 578.125 |
| 4000 | Tiempo excedido | Tiempo excedido | 33031.25 | 24718.75 | 2343.75 |
| 8000 | Tiempo excedido | Tiempo excedido | 183953.12 | 130078.125 | 9437.5 |
| 16000 | Tiempo excedido | Tiempo excedido | 859375.0 | 470906.25 | 37640.625 |
| 32000 | Tiempo excedido | Tiempo excedido | Tiempo excedido | 2’112512.25 | 165812.5 |
| 64000 | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido | 677781.25 |
| 128000 | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido |
| 256000 | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido |
| 512000 | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos |

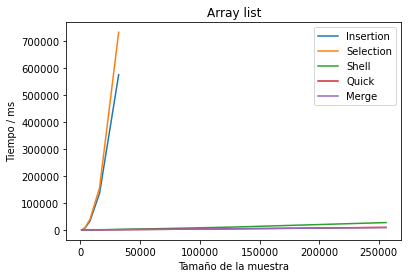


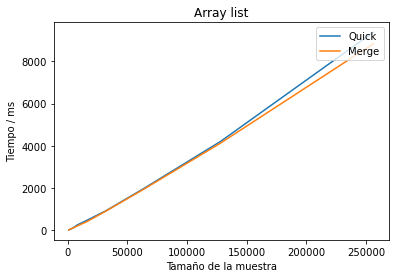


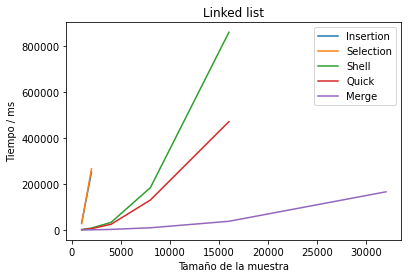












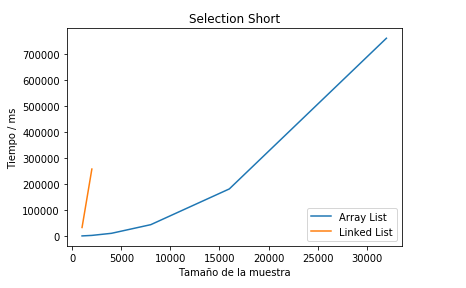
|  |  |  |
| --- | --- | --- |
| **Algoritmo** | **Array List** | **Linked List** |
| Insertion Sort | Mas eficiente |  |
| Selection Sort | Mas eficiente |  |
| Shell Sort | Mas eficiente |  |
| Quick Sort | Más eficiente |  |
| Merge Sort | Más eficiente |  |

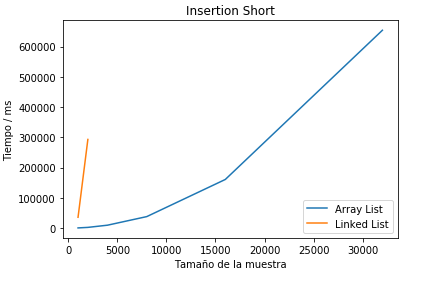
La mejor estructura de datos para este caso es Array List

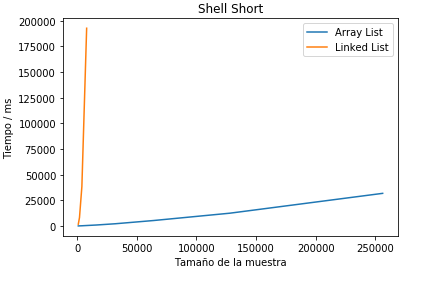
**Datos y gráficas maquina 2**

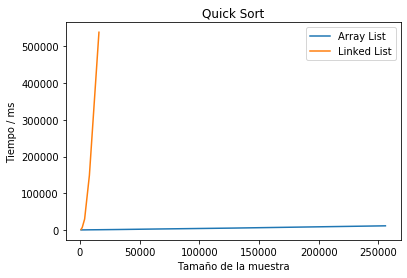
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (Array\_List)** | **Insertion Sort (ms)** | **Selection Sort (ms)** | **Shell Sort (ms)** | **Quick Sort (ms)** | **Merge Sort**  **(ms)** |
| 1000 | 578.12 | 687.5 | 31.25 | 31.25 | 15.625 |
| 2000 | 2328.12 | 2671.87 | 78.12 | 62.5 | 46.875 |
| 4000 | 9593.75 | 10734.37 | 171.87 | 125 | 109.375 |
| 8000 | 37968.75 | 44015.62 | 421.87 | 328.125 | 250 |
| 16000 | 160703.12 | 181640.62 | 906.25 | 546.875 | 515.625 |
| 32000 | 653781.25 | 761921.87 | 2140.62 | 1125 | 1093.75 |
| 64000 | Tiempo excedido | Tiempo excedido | 5281.25 | 2484.375 | 2343.75 |
| 128000 | Tiempo excedido | Tiempo excedido | 12437.5 | 5015.625 | 5281.25 |
| 256000 | Tiempo excedido | Tiempo excedido | 31781.25 | 11171.875 | 10625.0 |
| 512000 | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos |

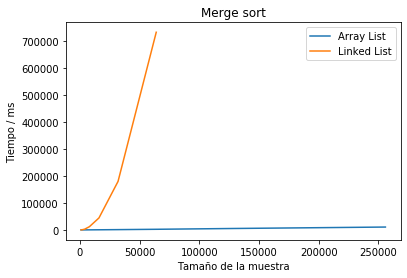
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (Linked\_List)** | **Insertion Sort (ms)** | **Selection Sort (ms)** | **Shell Sort (ms)** | **Quick Sort (ms)** | **Merge Sort**  **(ms)** |
| 1000 | 35796.87 | 33140.62 | 1734.37 | 1328.125 | 187.5 |
| 2000 | 292937.5 | 258156.25 | 8140.62 | 6703.125 | 703.125 |
| 4000 | Tiempo excedido | Tiempo excedido | 38343.75 | 29578.125 | 2734.375 |
| 8000 | Tiempo excedido | Tiempo excedido | 192765.62 | 148828.125 | 11953.125 |
| 16000 | Tiempo excedido | Tiempo excedido | Tiempo excedido | 538828.125 | 44265.625 |
| 32000 | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido | 179640.625 |
| 64000 | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido | 731953.125 |
| 128000 | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido |
| 256000 | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido | Tiempo excedido |
| 512000 | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Muestra excede el tamaño de los datos | Tiempo excedido | Tiempo excedido |

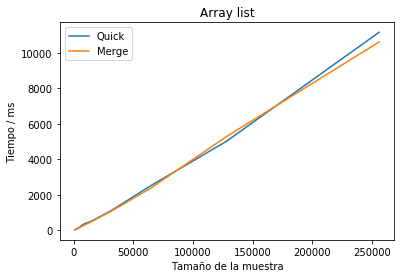
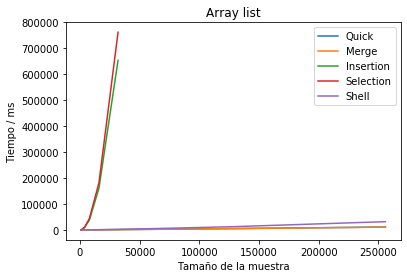


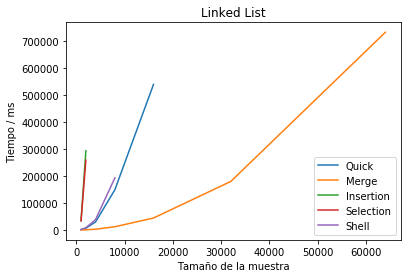












|  |  |  |
| --- | --- | --- |
| **Algoritmo** | **Array List** | **Linked List** |
| Insertion Sort | Más eficiente |  |
| Selection Sort | Más eficiente |  |
| Shell Sort | Más eficiente |  |
| Quick Sort | Más eficiente |  |
| Merge Sort | Más eficiente |  |

La mejor estructura de datos para este caso es Array List

**Preguntas de análisis**

1) ¿El comportamiento de los algoritmos es acorde a lo enunciado

teóricamente?

* Array list:

Tanto en quick sort como en merge sort se evidencia O(n log(n)), lo cual es el caso promedio.

* Linked list:

Al principio del merge sort si cumple con la forma determinada (O(n log(n))), sin embargo, al aumentar el tamaño de la muestra, empieza a tender a infinito debido a que sobrepasa el límite de los 720000 milisegundos determinado como tiempo excedido en este laboratorio. Por otro lado, el quick cumple la tendencia del peor caso(O(n^2)), es decir, cuadrático. No obstante, hay pocos datos ya que también empezara a sobrepasar el límite para tiempo excedido.

2) ¿Existe alguna diferencia entre los resultados obtenidos al ejecutar las

pruebas en diferentes máquinas?

No existen grandes diferencias en los tiempos de ejecución, debido a que la maquina 1 tiene tiempos levemente menores. Por lo tanto, la forma de las gráficas sea prácticamente igual.

3) De existir diferencias, ¿A qué creen ustedes que se deben dichas diferencias?

Las leves diferencias se deben a que la maquina 1 tiene el doble de memoria RAM que la maquina 2 y además que el procesador de la maquina 1 tiene más velocidad de procesamiento que la maquina 2, por estos dos factores se generan estas diferencias.

4) ¿Cuál Estructura de Datos es mejor utilizar si solo se tiene en cuenta los

tiempos de ejecución de los algoritmos?

Array list debido a que en todos los casos el tiempo de ejecución es menor sin importar el tipo de ordenamiento.

5) 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.

*Ranking:*

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