OBSERVACIONES DE LA PRACTICA

Santiago Castro Arciniegas Cod 202014994

Maria Camila Luna Velasco Cod 201920993

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | AMD Ryzen 7 3700U with Radeon Vega Mobile Gfx 2.30 GHz | AMD Ryzen 5 3500U with Radeon Vega Mobile Gfx 2.10 GHz |
| Memoria RAM (GB) | 8 GB | 8 GB |
| Sistema Operativo | Windows 10 Home 64 bits | Windows 10 Home 64 bits |

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) |
| 1000 | 953.125 | 1078.125 | 109.375 |
| 2000 | 4453.125 | 3703.125 | 140.625 |
| 4000 | 11671.875 | 14125.0 | 375.0 |
| 8000 | 51921.875 | 60078.125 | 593.75 |
| 16000 | 249921.875 | 249312.5 | 1234.375 |
| 32000 | Se demora mucho | Se demora mucho | 2640.625 |
| 64000 |  |  | 7781.25 |
| 128000 |  |  | 14875.0 |
| 256000 |  |  | 36953.125 |
| 375942 |  |  | 65953.125 |

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) |
| 1000 | 43921.875 | 35218.75 | 2140.625 |
| 2000 | 305821.25 | 309468.75 | 8781.25 |
| 4000 | Se demora mucho | Se demora mucho | 44734.375 |
| 8000 |  |  | 212031.25 |
| 16000 |  |  | Se demora mucho |
| 32000 |  |  |  |
| 64000 |  |  |  |
| 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) |
| Insertion sort | Es MUCHO mejor |  |
| Selection sort | Es MUCHO mejor |  |
| Shell sort | Es MUCHO mejor |  |

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.

# **Maquina 2**

## **Resultados**

|  |  |  |  |
| --- | --- | --- | --- |
| Tamaño de la muestra (ARRAY\_LIST) | Insertion Sort (ms) | Selection Sort (ms) | Shell Sort (ms) |
| 1000 | 1000.0 | 984.375 | 31.25 |
| 2000 | 4359.375 | 4250.0 | 109.375 |
| 4000 | 17531.25 | 17359.375 | 265.625 |
| 8000 | 75625.0 | 75312.0 | 656.25 |
| 16000 | 290750.0 | 315609.375 | 1640.625 |
| 32000 | Tarda mucho. | Tarda mucho. | 3687.5 |
| 64000 |  |  | 8734.375 |
| 128000 |  |  | 20796.875 |
| 256000 |  |  | 48062.5 |
| 375942 |  |  | 81156.25 |

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) |
| 1000 | 65421.875 | 34375.0 | 2187.5 |
| 2000 | 361953.125 | 301250.0 | 8609.375 |
| 4000 | Tarda mucho | Tarda mucho | 43656.25 |
| 8000 |  |  | 204656.25 |
| 16000 |  |  | Tarda mucho |
| 32000 |  |  |  |
| 64000 |  |  |  |
| 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) |
| Insertion sort | Es MUCHO mejor |  |
| Selection sort | Es MUCHO mejor |  |
| Shell sort | Es MUCHO mejor |  |

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.

# **Preguntas de análisis**

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

Sí, por un lado, el shellSort es mucho más eficiente que los otros dos algoritmos de ordenamiento; mientras que por otro al usar LinkedList el orden de crecimiento temporal es mucho mayor ya que los algoritmos de ordenamiento utilizan la función getElement que para esta Estructura de Datos es O(n), ya que tiene que recorrer toda la lista hasta hallar el elemento.

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

Las diferencias fueron mínimas ya que los computadores dl equipo son muy similares; sin embargo, hace más diferencia tener otras aplicaciones consumiendo memoria principal.

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

Habría diferencias si una máquina tuviera más memoria principal/RAM que la otra.

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

Es MUCHO mejor usar el arrayList, especialmente con shellSort porque este caso fue el único que logró ordenar todos los elementos (y en solo un minuto aprox.)

Observación Extra: En las gráficas de LinkedList no hay regresiones porque solo pudimos tomar dos datos de InsertionSort y SelectionSort (insuficiente información).