OBSERVACIONES DE LA PRACTICA

Javier Vargas Cod 202113305

Maicol Rincón Cod 202027329

# **Ambientes de pruebas**

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | Intel(R) Core(TM) i5-4300M CPU @ 2.60GHz, 2601 Mhz | Intel Core i3 – 3220 GHz |
| Memoria RAM (GB) | 16 | 4 Gb |
| Sistema Operativo | Microsoft Windows 10 Enterprise | Windows 10 Pro |

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

# **Maquina 1**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Porcentaje de la muestra [pct]** | **Tamaño de la muestra (ARRAYLIST)** | **Insertion Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 0.50% | 500 | 78,125 | 15,625 | 15,625 | 15,625 |
| 10.00% | 10000 | 343,75 | 31.25 | 46,875 | 62,5 |

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Porcentaje de la muestra [pct]** | **Tamaño de la muestra (LINKED\_LIST)** | **Insertion Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 0.50% | 500 | 109.375 | 46.875 | 46.875 | 109.375 |
| 10.00% | 2000 | 1.812.5 | 625.000 | 578.125 | 921.875 |

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

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| *Insertion Sort* | x |  |
| *Shell Sort* | x |  |
| *Merge Sort* | X |  |
| *Quick Sort* | x |  |

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

# **Maquina 2**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Porcentaje de la muestra [pct]** | **Tamaño de la muestra (ARRAYLIST)** | **Insertion Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 0.50% | 500 | 359.375 | 31.25 | 46.875 | 31.25 |
| 100.00% | 10000 | 39562.5 | 562.5 | 781.25 | 421.875 |

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

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Porcentaje de la muestra [pct]** | **Tamaño de la muestra (LINKED\_LIST)** | **Insertion Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 0.50% | 500 | 328.125 | 78.125 | 93.75 | 93.75 |
| 100.00% | 10000 | 39843.75 | 6281.25 | 6468.75 | 5984.375 |

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

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| *Insertion Sort* |  | X |
| *Shell Sort* | x |  |
| *Merge Sort* | X |  |
| *Quick Sort* | x |  |

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

# **Preguntas de análisis**

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

Si es acorde a lo visto en clase, sin embargo, tiene ciertas variaciones a la hora de ejecutar el programa

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

Si, los tiempos obtenidos en la maquina 1 son mejores a los de la maquina 2

1. De existir diferencias, ¿a qué creen que se deben?

Se debe a que la maquina 1 cuenta con mejores componentes que la maquina 2, también depende mucho de las tareas que este ejecutando la maquina a la hora de realizar las pruebas

1. ¿Cuál Estructura de Datos funciona mejor si solo se tiene en cuenta los tiempos de ejecución de los algoritmos? El arreglo tiene mejores tiempos de ejecución a medida que aumenta la cantidad de datos.

Merge Sort nos dio mejores tiempos, por lo tanto sería el mejor, y a medida que aumentan la cantidad de datos sentimos que los tiempos se comportan de manera igual, es decir no vemos una disminución de tiempo significativa con respecto al aumento de datos.

1. Teniendo en cuenta las pruebas de tiempo de ejecución por todos los algoritmos de ordenamiento estudiados (iterativos y recursivos), proponga un ranking de los mismo de mayor eficiencia a menor eficiencia en tiempo para ordenar la mayor cantidad de obras de arte.
2. Merge Sort
3. Quick Sort
4. Shell Sort
5. Selection Sort
6. Insertion Sort