OBSERVACIONES DE LA PRACTICA

Estudiante 1 Cod XXXX

Estudiante 2 Cod XXXX

# **Ambientes de pruebas**

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | AMD Ryzen 7 5700U with Radeon Graphics 1.80 GHz |  |
| Memoria RAM (GB) | 8 |  |
| Sistema Operativo | Windows 10 Pro 64-bits |  |

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% |  | 425,12 | 53,06 | 40,62 | 37,54 |
| 100.00% |  | 624902,08 | 4830,71 | 992,59 | 896,16 |

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% |  | 43306,82 | 1846,58 | 1815,94 | 185,75 |
| 100.00% |  | - | - | 212866,41 | 190458,29 |

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

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| *Insertion Sort* | 425,12 | 43306,82 |
| *Shell Sort* | 53,06 | 1846,58 |
| *Merge Sort* | 40,62 | 1815,94 |
| *Quick Sort* | 37,54 | 185,75 |

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% |  |  |  |  |  |
| 100.00% |  |  |  |  |  |

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% |  |  |  |  |  |
| 100.00% |  |  |  |  |  |

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

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

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?  
   La única incongruencia que se encontró en los tiempos de ordenamiento obtenidos fue la relación entre Quicksort y Mergesort, pues anque teóricamente estos dos algoritmos si deben ser los más rápidos, este primero debería ser más eficiente que el segundo, pues aplica el mismo concepto, pero subdividiendo aún más los datos. Sin embargo, los resultados obtenidos no reflejan esto, y en su lugar evidencian una mayor velocidad en el Mergesort.
2. ¿Existe alguna diferencia entre los resultados obtenidos al ejecutar las pruebas en diferentes máquinas?

Claramente existe una relación entre la velocidad de procesamiento y el equipo utilizado

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

La velocidad de procesamiento está estrechamente ligada a las especificaciones del equipo que desarrolla la tarea. Otras aplicaciones y tareas en segundo plano que puedan estar siendo ejecutadas también pueden generar alteraciones en los resultados.

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

La ARRAY\_LIST registró mejores tiempos de procesamiento, ya que es más sencillo navegar por los datos a través de una matriz

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.

* Quicksort
* Mergesort
* Shellsort
* Insertion sort