OBSERVACIONES DE LA PRACTICA

Estudiante 1 Cod 202111276

Estudiante 2 Cod 202116748

# **Ambientes de pruebas**

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | Intel(R) Core(TM) i5-10300H CPU @ 2.50GHz 2.50 GHz | Intel(R) Core(TM) i5-10310U CPU @ 1.70GHz 2.21 GHz |
| Memoria RAM (GB) | 8,00 GB | 16,0 GB |
| Sistema Operativo | Windows 10 Pro 64-bits | 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]** |
| Small | 768 datos | 1338.54 | 1119.79 | 1005.21 | 145.83 |
| 10.00% | 77 datos | 15.62 | 15.62 | 5.21 | 10.41 |

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]** |
| Small | 768 datos | 1348.96 | 1145.83 | 1031.25 | 145.83 |
| 10.00% | 77 datos | 15.62 | 15.62 | 5.21 | 10.41 |

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 | 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 (ARRAY\_LIST)** | **Insertion Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| Small | 768 datos | 2526.04 | 2104.16 | 1947.9 | 265.6 |
| 10.00% | 77 datos | 15.6 | 0.0 | 15.6 | 0.0 |

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]** |
| Small | 768 datos | 2489.5 | 2132.2 | 1953.12 | 265.6 |
| 10.00% | 77 datos | 15.6 | 15.6 | 15.6 | 15.6 |

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?

R/ Si es verdad lo esperado teóricamente, ya que el ARRAY\_LIST es más eficiente en comparación con el LINKED\_LIST.

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

R/ La máquina 2 fue ligeramente más rápida que la maquina 1.

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

R/ Puede que se por la diferencia en las memorias RAM.

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

R/ Según los tiempos de ejecución la estructura que funciona mejor es ARRAY\_LIST

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.

R/ La mejor es Merge Sort ya que tiene mayor eficiencia en relación al tiempo de ejecución, luego le sigue Quick Sort, después Shell Sort y por último Insertion Sort.