OBSERVACIONES DE LA PRACTICA

Estudiante 1 Cod XXXX

Estudiante 2 Cod XXXX

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | Intel(R) Core(TM) i5-3230 M Cpu @2.60GHZ | AMDRyzen 5 3550h fx 2.10Ghz |
| Memoria RAM (GB) | 6 | 8.00 |
| Sistema Operativo | Windows10x 64bits | Windows 10 x64 bits |

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

# **Maquina 1**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (ARRAYLIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 1171.875 | 1203.12 | 46.88 | 93.75 | 78.125 |
| 2000 | 4781.25 | 4765.625 | 140.63 | 156.25 | 140.625 |
| 4000 | 19609.375 | 20171.825 | 343.75 | 390.625 | 328.125 |
| 8000 | 78671.875 | 90390 | 843.75 | 875.0 | 859.375 |
| 16000 | 338000 | 345000 | 1796.88 | 1843.75 | 1921.875 |
| 32000 | >15mins | >15mins | 4656.25 | 4359.375 | 4265.625 |
| 64000 |  |  | 10062.5 | 10437.5 | 10234.375 |
| 128000 |  |  | 24031.5 | 24468.75 | 24328.125 |
| 256000 |  |  | 62140.625 | 63828.125 | 64687.5 |
| 375000 |  |  | 100312.8 | 101484.375 | 101984.375 |

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]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 108375 | 115406 | 5968.75 | 6468.75 | 6328.125 |
| 2000 | >15mins | >15mins | 31564.875 | 29031.25 | 29187.5 |
| 4000 |  |  | 166781.25 | 135718.75 | 132093.75 |
| 8000 |  |  | 808625 | 770875.0 | 709140.625 |
| 16000 |  |  | >15mins | >15mins | >15mins |
| 32000 |  |  |  |  |  |
| 64000 |  |  |  |  |  |
| 128000 |  |  |  |  |  |
| 256000 |  |  |  |  |  |
| 375000 |  |  |  |  |  |

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

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| Merge sort | x |  |
| Quick sort | x |  |

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.
  + Comparación de rendimiento para MergeSort.
  + Comparación de rendimiento para QuickSort.

# **Maquina 2**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (ARRAYLIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 43578 | 640 | 46 | 2484 | 2296 |
| 2000 | 355734 | 2593 | 78 | 10656 | 11437 |
| 4000 | 2890953 | 11781 | 187 | 49421 | 49500 |
| 8000 |  | 49014 | 453 | 250031 | 252531 |
| 16000 |  | 204531 | 984 | 1196103 | 1193390 |
| 32000 |  |  | 2375 |  |  |
| 64000 |  |  | 5734 |  |  |
| 128000 |  |  |  |  |  |
| 256000 |  |  |  |  |  |
| 512000 |  |  |  |  |  |

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]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 46484 | 40734 | 2296 | 2296 | 2328 |
| 2000 | 373434 | 328031 | 10656 | 11093 | 11321 |
| 4000 | 27389882 | 638703 | 49203 | 50531 | 50781 |
| 8000 |  |  | 247203 | 298578 | 264109 |
| 16000 |  |  | 1120781 | 1329921 | 1290734 |
| 32000 |  |  |  |  |  |
| 64000 |  |  |  |  |  |
| 128000 |  |  |  |  |  |
| 256000 |  |  |  |  |  |
| 512000 |  |  |  |  |  |

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

|  |  |  |
| --- | --- | --- |
| Algoritmo | Arreglo (ARRAYLIST) | Lista enlazada (LINKED\_LIST) |
| Merge sort | x |  |
| Quick sort | x |  |

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.
  + Comparación de rendimiento para MergeSort.
  + Comparación de rendimiento para QuickSort.

# **Preguntas de análisis**

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

Efectivamente, el tiempo de ejecución de el algoritmo Quicksort y mergesort son muy similares, y a su vez no hay mucha diferencia con el tiempo de ejecución del algoritmo Shell. Tambien los cambios en cada una de las tomas de datos nos pueden indicar que su comportamiento sigue la función (n log n)

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

Si existen diferencias

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

Probablemente se debe a la diferencia de procesadores de cada computador, los nucleos que tenga, la memoria Ram, o la cantidad de archivos abiertos al mismo tiempo

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

Indudablemente, los array lists son mucho mejores en el tiempo de ejecución que los single linked list

1. 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.

Array list:

1.) Shell sort

2.) Quick sort

3.) Merge sort

4.) Insertion sort

5.) Selection sort

Single linked list:

1.)Merge Sort

2.) Quick Sort

3.) Shell sort

4.)Insertion sort

5.) Selection sort