OBSERVACIONES DE LA PRACTICA

David Almanza - 202011293

Laura Daniela Arias Flórez - 202020621

# Máquina 1 Máquina 2

|  |  |  |
| --- | --- | --- |
| **Procesadores** | Intel Core i7-6700K  @4.00GHz | AMD Ryzen 5 3500U  @2.10 GHz |
| **Memoria RAM (GB)** | 16.0 GB | 12.0 GB |
| **Sistema Operativo** | Windows 10 Pro  64-bit | Windows 10 Home  64-bit |

*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) | Quick Sort (ms) | Merge Sort (ms) |
| 1000 | 609.38 | 640.63 | 31.25 | 31.25 | 15.63 |
| 2000 | 2921.87 | 2765.625 | 78.125 | 46.88 | 78.13 |
| 4000 | 11046.87 | 12937.5 | 171.875 | 171.88 | 125.0 |
| 8000 | 48265.63 | 46828.13 | 421.88 | 296.88 | 359.345 |
| 16000 | 206593.75 | 185343.75 | 1031.25 | 593.75 | 656.25 |
| 32000 | 748359.375 | 835078.125 | 2312.5 | 1328.13 | 1359.38 |
| 64000 |  |  | 5437.5 | 2765.63 | 2843.75 |
| 128000 |  |  |  | 6125.0 | 5812.5 |
| 256000 |  |  | 37656.25 | 13625.0 | 12562.5 |

*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 | 53656.25 | 55687.5 | 2531.25 | 1921.88 | 265.63 |
| 2000 | 491031.25 | 435515 | 14062.5 | 10875.00 | 1203.13 |
| 4000 | 3460203.13 | 3571750.0 | 58578.13 | 50187.50 | 4906.25 |
| 8000 |  |  |  | 189671.88 | 19250.0 |
| 16000 |  |  |  | 782109.38 | 71390.63 |
| 32000 |  |  |  |  |  |
| 64000 |  |  |  |  |  |
| 128000 |  |  |  |  |  |
| 256000 |  |  |  |  |  |

*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

# Maquina 2

## Resultados

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Tamaño de la muestra (ARRAY\_LIST) | Insertion Sort (ms) | Selection Sort (ms) | Shell Sort (ms) | Quick Sort  (ms) | Merge Sort (ms) |
| 1000 | 906.25 | 973.50 | 88.54 | 67.71 | 41.67 |
| 2000 | 3651.04 | 3567.71 | 192.71 | 140.63 | 78.13 |
| 4000 | 14911.46 | 14895.83 | 395.83 | 317.71 | 171.88 |
| 8000 | 63177.08 | 65828.13 | 1026.04 | 593.75 | 364.58 |
| 16000 | 249260.42 | 284911.46 | 2359.38 | 1125.00 | 796.88 |
| 32000 | 1037869.79 | 1090411.46 | 5244.79 | 1630.21 | 1697.92 |
| 64000 | +30 min | +30 min | 8495.63 | 3416.67 | 3562.50 |
| 128000 | -- | -- | 31187.50 | 7348.96 | 7838.54 |
| 256000 | -- | -- | 77869.79 | 17260.42 | 16479.67 |

*Tabla 5. 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 | 124432.29 | 105270.83 | 6380.21 | 2739.58 | 406.25 |
| 2000 | 630760.42 | 603000.00 | 25911.46 | 13953.13 | 1505.21 |
| 4000 | +30 min | +30 min | 118604.17 | 59822.92 | 5791.67 |
| 8000 | -- | -- | 583630.21 | 232057.29 | 22520.83 |
| 16000 | -- | -- | +30 min | +15 mins | 89312.50 |
| 32000 | -- | -- | -- | -- | 356932.29 |
| 64000 | -- | -- | -- | -- | +15 mins |

*Tabla 6. 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 7. Comparación de eficiencia de acuerdo con los algoritmos de ordenamientos y estructuras de datos utilizadas.*

## Graficas

# Preguntas de análisis

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

Así es. Como era de esperarse, Merge y Quick, ordenamientos de tipo recursivo, fueron las más efectivas para organizar los datos. Además, una vez más, los arreglos tuvieron un mejor tiempo que las listas enlazadas.

En la teoría, los órdenes de crecimiento para Quick y Merge son en su mayoría de carácter O(nlog(n)), a excepción del peor caso de Quick sort, el cual es de O(n2). Claramente, al no contar Excel con una línea de tendencia linearítmica, el análisis es más de tipo visual. Por un lado, en la gráfica de Quick podemos apreciar para la lista enlazada un comportamiento que aparenta ser de tipo O(n2), el cual, como se mencionó anteriormente, corresponde al peor caso de este ordenamiento. Para el arreglo, aunque no es muy discernible el tipo de crecimiento, al ser bastante efectivo, podemos concluir que debe tratarse de un crecimiento linearítmico. Por el otro lado, aunque las líneas de la gráfica de Merge se ven extremadamente diferentes, en ambas se puede distinguir la tendencia que muestra una gráfica xln(x) en un graficador. Hay que tener en cuenta que el tipo de estructura de datos también influencia los tiempos de los ordenamientos.

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

Sí, existen diferencias. La máquina 1 tuvo menores tiempos que la máquina 2 en general

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

A las características del equipo. El dispositivo de mejor procesador y sistema operativo, y más RAM pudo procesar la información más rápido

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

ARRAY\_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.**
   1. Merge
   2. Quick
   3. Shell
   4. Selection, Insertion