OBSERVACIONES DE LA PRACTICA

Estudiante 1: Nicole Murillo Fonseca 202025521

Estudiante 2: Mariana Diaz Arenas 202020993

|  |  |  |
| --- | --- | --- |
|  | Máquina 1 | Máquina 2 |
| Procesadores | 2,3 GHz Dual-Core Intel Core i5 | 1,6 GHz Intel Core i5 |
| Memoria RAM (GB) | 8 GB 2133 MHz LPDDR3 | 8 GB 1600 MHz DDR3 |
| Sistema Operativo | macOS Catalina Version 10.15.6 | macOs Sierra Version 10.12.6 |

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

TE = Tiempo exagerado > 30 minutos

Las gráficas están en escala logarítmica

*\*La valoración en la Tabla 4 puede contener inconsistencias debido a que la cantidad de datos recolectados no es igual en ambas tablas, pues al ejecutar los ordenamientos iterativos en LINKED\_LIST, los tiempos eran exagerados y no fue posible continuar con las pruebas. Debido a esto, concluimos que ARRAY\_LIST es más eficiente.*

# **Maquina 1**

## **Resultados**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tamaño de la muestra (ARRAYLIST)** | **Insertion Sort [ms]** | **Selection Sort [ms]** | **Shell Sort [ms]** | **Quick Sort [ms]** | **Merge Sort [ms]** |
| 1000 | 563,12 | 645,57 | 36,70 | 29,39 | 28,74 |
| 2000 | 2293,01 | 2593,07 | 77,56 | 58,77 | 77,22 |
| 4000 | 9248,66 | 10749,37 | 171,08 | 118,51 | 113,03 |
| 8000 | 38467,51 | 45474,61 | 417,92 | 295,15 | 239,77 |
| 16000 | 164240,64 | 190602,47 | 913,15 | 546,61 | 523,38 |
| 32000 | 688699,79 | 749434,38 | 2158,07 | 1124,42 | 1111,76 |
| 64000 | TE | TE | 5339,50 | 2386,79 | 2392,30 |
| 128000 | TE | TE | 12081,23 | 5017,01 | 5038,69 |
| 256000 | TE | TE | 31470,36 | 11219,67 | 10944,90 |
| 512000 | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria |

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 | 44019,46 | 39137,78 | 2196,17 | 1683,83 | 215,55 |
| 2000 | 362439,99 | 321500,04 | 10308,12 | 8180,27 | 837,87 |
| 4000 | 2996884,79 | 2615806,89 | 48337,59 | 35358,61 | 3349,79 |
| 8000 | TE | TE | 241678,25 | 184296,38 | 13507,21 |
| 16000 | TE | TE | 1075527,61 | 665179,07 | 55518,55 |
| 32000 | TE | TE | TE | TE | 221963,72 |
| 64000 | TE | TE | TE | TE | 900275,62 |
| 128000 | TE | TE | TE | TE | TE |
| 256000 | TE | TE | TE | TE | TE |
| 512000 | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria |

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 | 988,85 | 943,70 | 55,39 | 40,70 | 40,30 |
| 2000 | 3867,08 | 3827,65 | 117,87 | 83,61 | 77,72 |
| 4000 | 15574,41 | 15657,51 | 243,85 | 164,84 | 160,47 |
| 8000 | 55075,15 | 64752,74 | 606,97 | 428,51 | 366,56 |
| 16000 | 241379,10 | 273979,95 | 1330,29 | 808,07 | 767,66 |
| 32000 | 919653,80 | 1088104,77 | 3122,55 | 1627,12 | 1557,04 |
| 64000 | TE | TE | 7263,97 | 3346,02 | 3353,67 |
| 128000 | TE | TE | 17628,31 | 7014,57 | 7107,16 |
| 256000 | TE | TE | 45064,69 | 15330,44 | 15462,08 |
| 512000 | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria |

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 | 65290,13 | 57201,13 | 3543,36 | 2543,69 | 312,12 |
| 2000 | 526327,23 | 466274,57 | 20680,97 | 12049,21 | 1284,75 |
| 4000 | TE | TE | 81845,15 | 54110,67 | 5060,59 |
| 8000 | TE | TE | 351514,18 | 280243,23 | 20667,49 |
| 16000 | TE | TE | 1571672,10 | 1002307,04 | 81073,78 |
| 32000 | TE | TE | TE | TE | 331366,32 |
| 64000 | TE | TE | TE | TE | 1330258,06 |
| 128000 | TE | TE | TE | TE | TE |
| 256000 | TE | TE | TE | TE | TE |
| 512000 | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria | Excede el tamaño en memoria |

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?

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

Sí. Haciendo una comparativa entre los tiempos finales indicados en las tablas de la máquina 1 y 2, se puede evidenciar que la máquina 2, tuvo tiempos de ejecución notablemente mayores en todas las muestras y ordenamientos (iterativos y recursivos) que la máquina 1.

3) De existir diferencias, ¿A qué creen ustedes que se deben dichas diferencias?

Consideramos que la diferencia en tiempos de ejecución se debe al procesador y al sistema operativo de las máquinas, pues la máquina 1 es un MacBook Pro con un procesador de 2,3 GHz y un sistema operativo más reciente (macOS Catalina); mientras que, la máquina 2 es un MacBook Air con un procesador de 1,6 GHz y un sistema operativo que no es el más reciente (macOs Sierra). Ahora bien, es posible que la memoria RAM también haya sido un factor determinante en los tiempos dispares, pues aunque ambas máquinas tiene 8GB de ram, la máquina 2 tiene una RAM de 8 GB 1600 MHz DDR3, mientras que la máquina 1 tiene una versión más reciente con una RAM de 8 GB 2133 MHz LPDDR3.

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

Teniendo en cuenta los tiempos de ejecución es claro que la Estrcutura de Datos más óptima es el ARRAY\_LIST o el arreglo, pues los tiempos fueron significativamente menores, y por tanto, el algoritmo fue más eficiente.

5) 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.

Mayor eficiencia en tiempo

1. Merge Sort
2. Quick Sort
3. Shell Sort
4. Selection Sort
5. Insertion Sort

Menor eficiencia en tiempo