OBSERVACIONES DE LA PRACTICA

María Alejandra Estrada García Cod. 202021060

Santiago Martínez Novoa Cod. 202112020

# **Ambientes de pruebas**

|  |  |  |
| --- | --- | --- |
|  | **Máquina 1** | **Máquina 2** |
| **Procesadores** | Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz | Intel(R) Core(TM) i3-8145U CPU @ 2.10GHz 2.30 GHz |
| **Memoria RAM (GB)** | 8.00 GB (7.82 GB utilizable) | 8,00 GB (7,89 GB utilizable) |
| **Sistema Operativo** | Windows 10 64-bit x64-based processor | Windows 10 64 bits, procesador x64 |

# *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% | -small(768) | 31.25 | 41,67 | 36,45 | 26,04 |
| 10.00% | -10pct(15008) | 390.62 | 932.3 | 2432,3 | 486.38 |

*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% | -small(768) | 916,67 | 1010,42 | 994.8 | 161.46 |
| 10.00% | -10pct(15008) | 344234.38 | 866421.88 | 1772062.5 | 36281.25 |

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

|  |  |  |
| --- | --- | --- |
| **Algoritmo** | **Arreglo (ARRAYLIST)** | **Lista enlazada (LINKED\_LIST)** |
| *Insertion Sort* | El primero más eficiente | El segundo más eficiente |
| *Shell Sort* | El tercer más eficiente | El tercero más eficiente |
| *Merge Sort* | El segundo más eficiente | El primero más eficiente |
| *Quick Sort* | El cuarto más eficiente | El cuarto más eficiente |

*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% | -small(768) | 25.0 | 23.44 | 34.38 | 27.34 |
| 10.00% | -10pct(15008) | 463.54 | 1393.84 | 3874.75 | 645.84 |

*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% | -smal(768) | 1244.79 | 1463.55 | 1380.21 | 177.09 |
| 10.00% | -10pct(15008) | 299328.13 | 1767703.13 | 3842760.13 | 76140.63 |

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

|  |  |  |
| --- | --- | --- |
| **Algoritmo** | **Arreglo (ARRAYLIST)** | **Lista enlazada (LINKED\_LIST)** |
| *Insertion Sort* | El primero más eficiente | El segundo más eficiente |
| *Shell Sort* | El tercer más eficiente | El tercero más eficiente |
| *Merge Sort* | El segundo más eficiente | El primero más eficiente |
| *Quick Sort* | El cuarto más eficiente | El cuarto más eficiente |

*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// Como se explicó en la clase, cada tipo de ordenamiento es diferente, en la forma de ordenar los datos con los distintos algoritmos. Se pudo demostrar que el ‘ARRAY\_LIST’ es más rápido que el ‘LINKED\_LIST’, por lo que cambiará el orden de crecimiento temporal de las operaciones. Sin embargo, funcionan diferentes pues en el single\_linked\_list es más eficiente con el merge sort, pero en la maquina 1 y 2, el insertion sort fue más eficiente en tiempo al usar “Array\_list”, al comparar con las dos TAD listas.

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

R// Si, se presentaron diferentes tiempos de procesamientos. Es decir, en el rendimiento de los algoritmos, debido a la diferencia de los procesadores en cada máquina. La máquina 1 tiene un procesador Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz, mientras que la máquina 2 tiene un procesador Intel(R) Core(TM) i3-8145U CPU @ 2.10GHz 2.30 GHz.

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

R// Como se puede observar por medio de las tablas de comparación (de tiempos de ejecución de los diferentes algoritmos) y los ordenamientos, es el tiempo en que toma las dos máquinas en ejecutarlos. Se muestra que en la máquina 2 dura menos tiempo cuando se utilizan menor cantidad de datos con ‘ARRAY\_LIST’, sin embargo, cuando se utiliza una mayor cantidad de datos la máquina 1 tiene mejor tiempo de rendimiento, en especial al utilizar el TAD lista ‘LINKED\_LIST’. Las principales diferencias de las características de los procesadores se encuentran en el número de núcleos e hilos de proceso y su velocidad (máquina 1: Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz, máquina 2: Intel(R) Core(TM) i3-8145U CPU @ 2.10GHz 2.30 GHz). Las velocidades (GHz) de los procesadores fueron distintas, siendo la mayor diferencia. Es decir, que la máquina 2 tiene mayor frecuencia de funcionamiento, en comparación a la 1. Por otro lado, máquina 2 tiene un Intel(R) Core de menor nivel. Por otro lado, la máquina 1 tiene menor memoria RAM utilizable, en comparación con la segunda. Al realizar las pruebas de rendimiento se encontró que, aunque en los primeros intentos la maquina 2 tuvo en algunos casos mejor rendimiento que la maquina 1, luego del promedio la máquina 1 fue siempre superior. Esto puede ser causa de la consistencia en el procesamiento del i5 frente a los rendimientos inesperados del i3.

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

R// El ‘ARRAY\_LIST’ funciona más rápido en ambas computadoras debido a que en los arreglos almacenan elementos en ubicaciones de memoria contiguas, lo que permite un acceso más rápido a un elemento en un índice específico. Por otro lado, debido a que, en la lista enlazada, los elementos no se almacenan en ubicaciones contiguas, por lo que deben almacenarse con etiquetas adicionales que den una referencia al siguiente elemento, y toma más tiempo.

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// Teniendo en cuenta las pruebas realizadas con las diferentes máquinas con el tiempo de ejecución de los algoritmos de ordenamientos –iterativos y recursivos- el ranking de menor a mayor eficiencia de tiempo (teniendo en cuenta los dos tipos de TAD Listas) es el siguiente:

1. Insertion sort
2. Merge sort
3. Shell sort
4. Quick sort