**REPRESENTACIÓN ARREGLO**

**Tabla de complejidad temporal teórica en el peor caso en representación Arreglo**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Insertion Sort** | **Shell Sort** | **Merge Sort** | **Quick Sort** |
| **Complejidad O (…)**  **en el peor caso para una muestra de N datos** |  |  |  |  |

**Tabla de rendimiento para los distintos algoritmos de ordenamiento en Arreglo**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tamaño de la muestra** | **Insertion Sort**  **(tiempos en ms)** | **Shell Sort**  **(tiempos en ms)** | **Merge Sort**  **(tiempos en ms)** | **Quick Sort**  **(tiempos en ms)** |
| **1000** | 830 ms | 58 ms | 12 ms | 41 ms |
| **2000** | 6278 ms | 180 ms | 23 ms | 136 ms |
| **4000** | 50466 ms | 866 ms | 59 ms | 561 ms |
| **8000** | 387249 ms | 3828 ms | 260 ms | 2319 ms |
| **16000** | excede | 17416 ms | 883 ms | 10045 ms |
| **32000** | excede | 79028 ms | 3550 ms | 43352 ms |
| **64000** | excede | 402739 ms | 14429 ms | 176046 ms |
| **128000** | excede | excede | 60917 ms | excede |
| **256000** | excede | excede | 292924 ms | excede |
| **512000** | excede | excede | excede | excede |

**Tabla de comparación de algoritmos de ordenamiento en Arreglo**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Algoritmo más eficiente en tiempo** | **Segundo algoritmo más eficiente en tiempo** | **Tercer algoritmo más eficiente en tiempo** | **Cuarto algoritmo más eficiente en tiempo** |
| **Algoritmo** | *MergeSort* | *QuickSort* | *ShellSort* | *InsertionSort* |

**REPRESENTACIÓN LISTA ENLAZADA**

**Tabla de rendimiento para los distintos algoritmos de ordenamiento en Lista Enlazada**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tamaño de la muestra** | **Insertion Sort**  **(tiempos en ms)** | **Shell Sort**  **(tiempos en ms)** | **Merge Sort**  **(tiempos en ms)** | **Quick Sort**  **(tiempos en ms)** |
| **1000** | 815 ms | 58 ms | 12 ms | 36 ms |
| **2000** | 6480 ms | 186 ms | 22 ms | 137 ms |
| **4000** | 51974 ms | 825 ms | 66 ms | 581 ms |
| **8000** | 393687 ms | 3923 ms | 221 ms | 2287 ms |
| **16000** | excede | 18186 ms | 851 ms | 10319 ms |
| **32000** | excede | 79989 ms | 3522 ms | 43417 ms |
| **64000** | excede | 404512 ms | 13926 ms | 177707 ms |
| **128000** | excede | excede | 58942 ms | excede |
| **256000** | excede | excede | 319991 ms | excede |
| **512000** | excede | excede | excede | excede |

**Tabla de comparación de algoritmos de ordenamiento en Lista Enlazada**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Algoritmo más eficiente en tiempo** | **Segundo algoritmo más eficiente en tiempo** | **Tercer algoritmo más eficiente en tiempo** | **Cuarto algoritmo más eficiente en tiempo** |
| **Algoritmo** | *MergeSort* | *QuickSort* | *ShellSort* | *InsertionSort* |

**COMPARACIÓN DE REPRESENTACIONES**

**Tabla de comparación de cada algoritmo de ordenamiento de acuerdo con la Estructura de Datos utilizada (Arreglo y Lista Enlazada)**

|  |  |  |
| --- | --- | --- |
|  | **Estructura de Datos más eficiente en tiempo** | **Estructura de Datos menos eficiente en tiempo** |
| **Insertion Sort** | *Arreglo Dinámico* | *Lista Enlazada* |
| **Shell Sort** | *Arreglo Dinámico* | *Lista Enlazada* |
| **Merge Sort** | *Arreglo Dinámico* | *Lista Enlazada* |
| **Quick Sort** | *Arreglo Dinámico* | *Lista Enlazada* |

**Pregunta 1. A partir de la tabla anterior, en el caso general de ordenamiento. ¿Cuál Estructura de Datos es mejor utilizar si solo se tiene en cuenta los tiempos de ejecución de los algoritmos?**

Para todos los algoritmos el usar un arreglo dinámico daba una respuesta un poco más rápida, sin embargo, en todos los casos la diferencia no era inmensa y por ende en algunos ordenamientos con ciertas muestras la lista enlazada llegaba a ser un poco más rápida que el arreglo; gracias a esto se podría concluir que no es muy relevante escoger una u otra estructura de datos si solo se tiene en cuenta el tiempo de ejecución de los algoritmos. De todas formas, cabe resaltar que donde sí había una diferencia considerable era al momento de crear sublistas que servirían como muestras para probar los algoritmos, ya que en la lista enlazada el tiempo era mucho mayor.