|  |  |
| --- | --- |
| **Nombre del Algoritmo** | ShellSort |
| **Mejor caso** | Este algoritmo presenta su mejor caso cuando el conjunto inicial de datos se encuentra casi completamente ordenado. |
| **Complejidad en el mejor caso (notación O)** | O() |
| **Peor caso** | Este algoritmo presenta su peor caso cuando el conjunto inicial de datos tiene posiciones aleatorias. |
| **Complejidad en el peor caso (notación O)** | O() |
| **Algoritmo *Inplace*** | Si |
| **Algoritmo Adaptativo** | Si |
| **Algoritmo Estable** | No |

**TALLER 3**

**DOCUMENTO DE CARACTERÍSTICAS ANÁLISIS DE ALGORITMOS DE ORDENAMIENTO**

**A. RESUMEN DE INFORMACIÓN DE UN ALGORITMO DE ORDENAMIENTO:**

|  |  |
| --- | --- |
| **Nombre del Algoritmo** | MergeSort |
| **Mejor caso** | Dado que el algoritmo no es adaptativo, en cualquier situación del conjunto de datos se demora lo mismo. |
| **Complejidad en el mejor caso (notación O)** | O() |
| **Peor caso** | Dado que el algoritmo no es adaptativo, en cualquier situación del conjunto de datos se demora lo mismo. |
| **Complejidad en el peor caso (notación O)** | O() |
| **Algoritmo *Inplace*** | No |
| **Algoritmo Adaptativo** | No |
| **Algoritmo Estable** | Si |

|  |  |
| --- | --- |
| **Nombre del Algoritmo** | QuickSort |
| **Mejor caso** | En este algoritmo existen 3 mejores casos que terminan en igual tiempo, cuando el conjunto inicial de datos se encuentra invertido, cuando el conjunto inicial de datos se encuentra ordenado aleatoriamente, y cuando el conjunto inicial de datos se encuentra casi completamente ordenado. |
| **Complejidad en el mejor caso (notación O)** | O() |
| **Peor caso** | En este algoritmo existen 2 peores casos; el primero, cuando el conjunto inicial de datos cuenta con muchos elementos del mismo tipo repetidos; el segundo cuando la lista se encuentra directamente ordenada. |
| **Complejidad en el peor caso (notación O)** | O() |
| **Algoritmo *Inplace*** | Si |
| **Algoritmo Adaptativo** | Si |
| **Algoritmo Estable** | No |

**B. TABLA DE COMPARACIÓN DE TIEMPOS DE LOS ALGORITMOS DE ORDENAMIENTO PARA EL CONJUNTO COMPLETO DE COMPARENDOS:**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **ShellSort (mseg)** | **MergeSort (mseg)** | **QuickSort (mseg)** |
| Tiempo Ejecución 1 | 437 | 155 | 577 |
| Tiempo Ejecución 2 | 484 | 149 | 547 |
| Tiempo Ejecución 3 | 500 | 147 | 552 |
| Tiempo Ejecución 4 | 500 | 140 | 547 |
| Tiempo Ejecución 5 | 506 | 146 | 638 |
| **Tiempo promedio (mseg)** | 485.4 | 147.4 | 572.2 |

**C. CONCLUSIÓN:**

Por el tiempo promedio de ejecución, para el caso general, el algoritmo más eficiente es MergeSort. El siguiente algoritmo en eficiencia es ShellSort. El algoritmo menos eficiente es QuickSort.

**ACLARACIÓN ADICIONAL:**

Todas las pruebas realizadas durante el taller 3 fueron hechas en los computadores de la sala de informática WAIRA, ubicada en el sexto piso del ML. Cualquier duda o inquietud respecto a los datos obtenidos, por favor realizar las mismas pruebas en WAIRA.

**BIBLIOGRAFÍA Y REFERENCIAS:**

* Sedgwick, R. (1988). Algoritms (4ª ed.). Londres, Inglaterra: Addison-Wesley.
* Tomic, J. (s.f.). Sorting Algorithms Animations. Recuperado 27 febrero, 2020, de https://www.toptal.com/developers/sorting-algorithms