|  |  |
| --- | --- |
| Nombre del Algoritmo | Shell |
| Mejor caso | Cuando los datos están ligeramente ordenados. |
| Complejidad del mejor caso | NLog3N |
| Peor caso | Cuando los datos están aleatorios. |
| Complejidad en el peor caso | c\*N^3/2 |
| Algoritmo inPlace | Sí |
| Algoritmo adaptativo | Sí |
| Algoritmo Estable | No |
|  |  |
| Nombre del Algoritmo | Merge |
| Mejor caso | Al dividir el arreglo en dos el tiempo es el mismo para todos los casos, sin embargo, la cantidad de intercambios puede cambiar, y en ese caso es que se hagan muy pocos intercambios. |
| Complejidad del mejor caso | 1/2 N\*Log N |
| Peor caso | Al dividir el arreglo en dos el tiempo es el mismo para todos los casos, sin embargo, la cantidad de intercambios puede cambiar, y en ese caso es que se hagan todos los intercambios. |
| Complejidad en el peor caso | N\*log N |
| Algoritmo inPlace | No |
| Algoritmo adaptativo | No |
| Algoritmo Estable | Sí |
|  |  |
| Nombre del Algoritmo | Quick |
| Mejor caso | Cualquiera que no sea no tenga muchos repetidos. |
| Complejidad del mejor caso | 1/2 N\*Log N |
| Peor caso | Cuando los datos tienen muchos repetidos. |
| Complejidad en el peor caso | 1/2\*N^2 |
| Algoritmo inPlace | No |
| Algoritmo adaptativo | No |
| Algoritmo Estable | Sí |

|  |  |  |  |
| --- | --- | --- | --- |
| Ordenamientos | ShellSort (mseg) | MergeSort (mseg) | QuickSort (mseg) |
| Tiempo Ejecución 1 | 19987 | 8084 | 16217 |
| Tiempo Ejecución 2 | 18707 | 7989 | 15470 |
| Tiempo Ejecución 3 | 20025 | 9301 | 16008 |
| Tiempo Promedio (mseg) | 19573 | 8458 | 15898 |

Con base en los datos obtenidos podemos concluir que el mejor ordenamiento para trabajar es MergeSort, seguido de QuickSort, y terminando con ShellSort.