ALGORITMOS I

Datos.
Nombre: Oscar Fernández Durán.

Descripción.

En el presente documento se muestra la implementación de tres algoritmos de ordenamiento, así como el comportamiento que presentan cada uno de ellos.
Los algoritmos que se implementaron se muestran a continuación y enseguida se da una breve descripción de cada uno de ellos:

Quicksort: este algoritmo, basado en la técnica divide y vencerás, consiste en elegir un elemento del conjunto de datos a ordenar y llamarlo pivote, después, tomarlo como referencia e ir poniendo los dema´s elementos del conjunto de tal manera que a un lado queden todos los elementos menores que él, y al otro los mayores, garantizando con esto que el pivote ocupe exactamente su lugar. Ahora nos hemos quedado con dos subconjuntos del conjunto original de datos, uno con los elementos menores que el pivote y otro con los elementos mayores, lo que procede ahora es realizar lo mismo que se hizo con el arreglo original, pero ahora con ambos subconjuntos mientras estos contengan m´as de un elemento, realizando esto de manera recursiva. Una vez que se termina este proceso todos los elementos se encuentran ordenados.

Merge sort: este algoritmo, de igual manera que el anterior, se basa en la t´ecnica divide y vencera´s, lo que este algoritmo realiza es, tomar el conjunto de datos y dividirlo en dos subconjuntos de aproximadamente el mismo taman˜o, despu´es, volver a aplicar este algoritmo de manera recursiva, pero ahora a los subconjuntos que se obtuvieron hasta que queden conjuntos de un solo elemento y mezclarlos comparando cada uno de los elementos de los subconjuntos y dejarlos ahora en un conjunto de datos ordenados.

Heapsort: este algoritmo, basado en una estructura de árbol binario, forma a partir de un conjunto de datos, un árbol donde en la raíz siempre se ubica el elemento mayor, denominado como padre, de esta manera, siempre debemos tener que el padre es mayor que los hijos, se va revisando esto para cada uno de los nodos del árbol y de no ser así se realiza el intercambio, una vez que el mayor ocupa la raíz se van obteniendo as´ı los elementos mayores del conjunto en cada una de las iteraciones y formando el conjunto de datos ordenado, para esto, el u´ltimo elemento del a´rbol queda en la cima y se realiza de nuevo el proceso para ir obteniendo así los elementos mayores del conjunto.

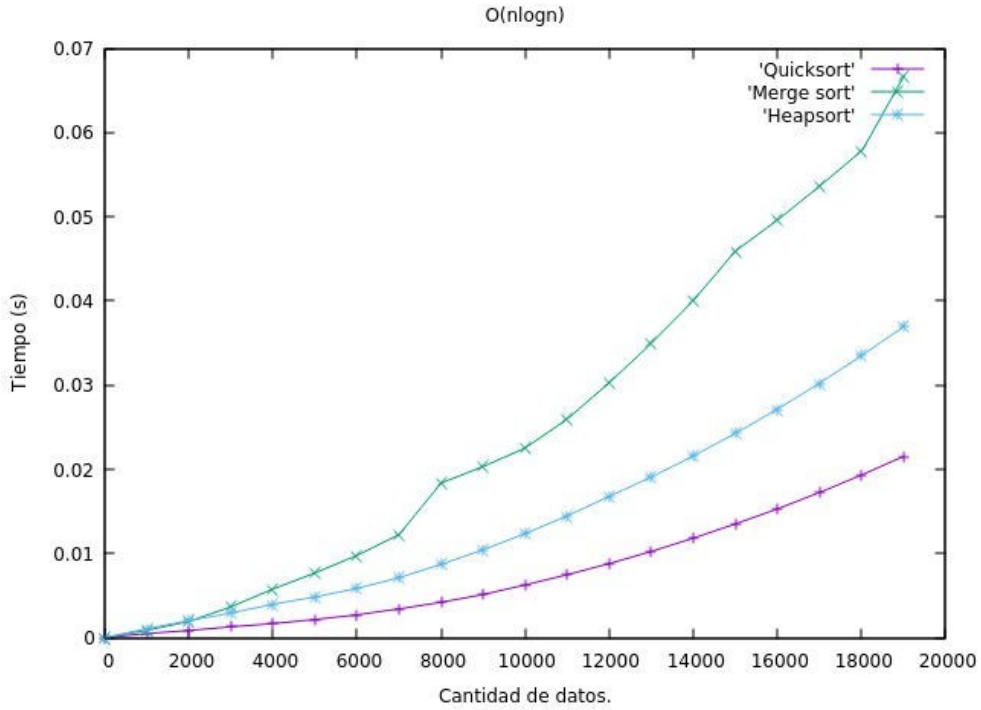

Figura 1: gráfico que muestra el comportamiento de los algoritmos descritos.

Conclusión.

En la Figura 1 podemos observar que el algoritmo más eficiente es el Quicksort, seguido por Heapsort y Merge sort respectivamente.

Si bien es claro que para una cantidad de datos pequeña cada uno tiene un tiempo de ejecución similar, en tanto la cantidad de datos va aumentando podemos notar que Merge sort se dispara, esto se debe a que Merge sort realiza la separación de los arreglos a la mitad, cosa que Quicksort se ahorra. Es importante notar que Quicksort en el peor de sus casos tiene un orden $O(n^2)$, pero para que esto suceda la lista deber´ıa estar ordenada ya que no habra´ elementos menores que él y siempre va a generar a su izquierda un conjunto de datos vacío, lo que es ineficiente, en el caso de Heapsort la diferencia se encuentra en la cantidad de cambios que realiza y que aunque los elementos est´en ordenados va a intercambiar todos los elementos para ordenar el conjunto, con esto podemos ver que un gran punto a favor de Quicksort es que no hace intercambios innecesarios de elementos.