Algoritmo de ordenamiento Quicksort

En el desarrollo de este software la primera etapa de elaboración se investigo el funcionamiento del algoritmo y se practico de forma manual para comprender de manera correcta y completa.

El funcionamiento del algoritmo se podría resumir de la siguiente manera.

* Elegir un elemento de la lista de elementos a ordenar, al que llamaremos **pivote**.
* Resituar los demás elementos de la lista a cada lado del pivote, de manera que a un lado queden todos los menores que él, y al otro los mayores. Los elementos iguales al pivote pueden ser colocados tanto a su derecha como a su izquierda, dependiendo de la implementación deseada. En este momento, el pivote ocupa exactamente el lugar que le corresponderá en la lista ordenada.
* La lista queda separada en dos sublistas, una formada por los elementos a la izquierda del pivote, y otra por los elementos a su derecha.
* Repetir este proceso de forma recursiva para cada sublista mientras éstas contengan más de un elemento. Una vez terminado este proceso todos los elementos estarán ordenados.

La elección del pivote es de suma importancia para la eficiencia del algoritmo, la cual se puede dividir en tres casos:

1. En el mejor caso, el pivote termina en el centro de la lista, dividiéndola en dos sublistas de igual tamaño. En este caso, el orden de complejidad del algoritmo es [O](https://es.wikipedia.org/wiki/Cota_superior_asint%C3%B3tica)(n·log n).
2. En el peor caso, el pivote termina en un extremo de la lista. El orden de complejidad del algoritmo es entonces de O(n²). El peor caso dependerá de la implementación del algoritmo, aunque habitualmente ocurre en listas que se encuentran ordenadas, o casi ordenadas. Pero principalmente depende del pivote, si por ejemplo el algoritmo implementado toma como pivote siempre el primer elemento del [array](https://es.wikipedia.org/wiki/Array), y el array que le pasamos está ordenado, siempre va a generar a su izquierda un array vacío, lo que es ineficiente.
3. En el caso promedio, el orden es O(n·log n).

Para la implementación del algoritmo se definió el pivote en la mitad de la lista y así poder obtener una implementación promedio y un costo de O(n·log n). No se definió una forma mas optima en la que se elije el pivote porque se le dio prioridad a la creación del algoritmo para valores primitivos y genéricos.

Se desarrollaron dos clases, una clase para primitivos, donde se ordenan valores integer almacenados en un arreglo INT [], y otra clase con valores genéricos donde se recibe un List genérico LIST <T> que se ordenan de igual manera como en los primitivos.