PROYECTO DE ESTRUCTURA
ALGORITMOS DE SORTING

El objetivo con este trabaja es usar distintos algoritmos de sorting para comparar su eficiencia en la practica basandose en los tiempos de los casos promedio de cada uno. Para llevar a cabo este proyecto se usaron 3 algoritmos de sorting: quick-sort, tree-sort y heap-sort, con dos implementaciones distintas para este ultimo, una con arreglos y otra con listas dinamicas.
Para los casos de prueba, se utilizara un arreglo previamente cargado con numeros aleatorios. La capacidad o cantidad de elementos a probar seran valores que se iran incrementando. Se probara con 10, 100, 1000, 10000, 100000, 1000000 elementos.

QUICKSORT

Se dice que el tiempo de este algoritmo de sorting depende de la implementacion del partition. Para este caso, se selecciona como pivot al primer elemento por lo que el peor caso hace que el partition sea orden N, pero en la practica se comporta como n * log n. Con este algoritmo fue posible ordenar una colección de 1 millon de elementos, obteniendo mejores resultados que en otros algoritmos.

TREESORT

Si bien en el peor caso este algoritmo es de orden N, en la practica se comporta como un algoritmo de orden n * Log n. Este algoritmo se comporto mas rapido que los algoritmos que usaban heaps en los casos con colecciones “pequeñas”, pero cuando nos acercamos a colecciones de 1 millon de elementos se puede notar que demora un poco mas.

HEAPSORT (ARREGLOS)

El peor tiempo de este algoritmo es de orden n * Log n, por lo que decimos que se comporta bastante bien en la practica. De las dos implementacions de heap, esta obtuvo mejores resultados. Si bien demora un poco mas que otros algoritmos para colecciones “pequeñas”, para casos mas grandes su tiempo mejora.

HEAPSORT (LISTAS DINAMICAS)

Tambien es un algoritmo de orden n * Log n, pero a diferencia de la implementacion con arreglos, usando listas dinamicas solo se pudo hacer que ordene colecciones de 10000 elementos de manera eficiente.

Resultados Obtenidos

La siguiente tabla de doble entrada expresa los resultados obtenidos para colecciones de elementos de distintos tamaños, ordenados con los distintos algoritmos de sorting. La tabla expresa el tiempo en magnitud de mili segundos.

	10	100	1000	10000	100000	1000000
QuickSort	0	1	7	14	71	1172
TreeSort	0	2	5	38	316	5508
HeapSort (array)	160	161	179	184	324	3886
HeapSort (dynamic)	0	3	68	5031	492536	