Universidad del Valle de Guatemala
AED
Sec. 10

Marco Ramirez 21032
Andres Quezada 21085
Francisco Castillo 21562

Eficiencia de los algoritmos de ordenamiento

Utilización de Profiler

Para el desarrollo de la hoja de trabajo, se ha utilizado el ambiente de desarrollo IntelliJ, el cual ya trae un profiler integrado. En la sección de ayuda del sitio oficial del IDE existen publicaciones oficiales que demuestran el funcionamiento de la herramienta. A través del análisis del reporte generado se han obtenido los tiempos de corrida de las funciones.

Resultados

Los resultados subrayados en rojo se deben a que el GnomeSort no respondía a en las pruebas a partir de los 500 datos (en cantidades menores a 100 si funciona el algoritmo y si pasó los junit), por lo que se consultó con Moises y acordamos dejarlo indicado en la entrega de manera que no existen datos para ello, ya que es algo fuera de nuestro control el hecho de la falta de respuesta, probamos esperar hasta 1 hora y no retornaba nada.Gracias por su comprensión

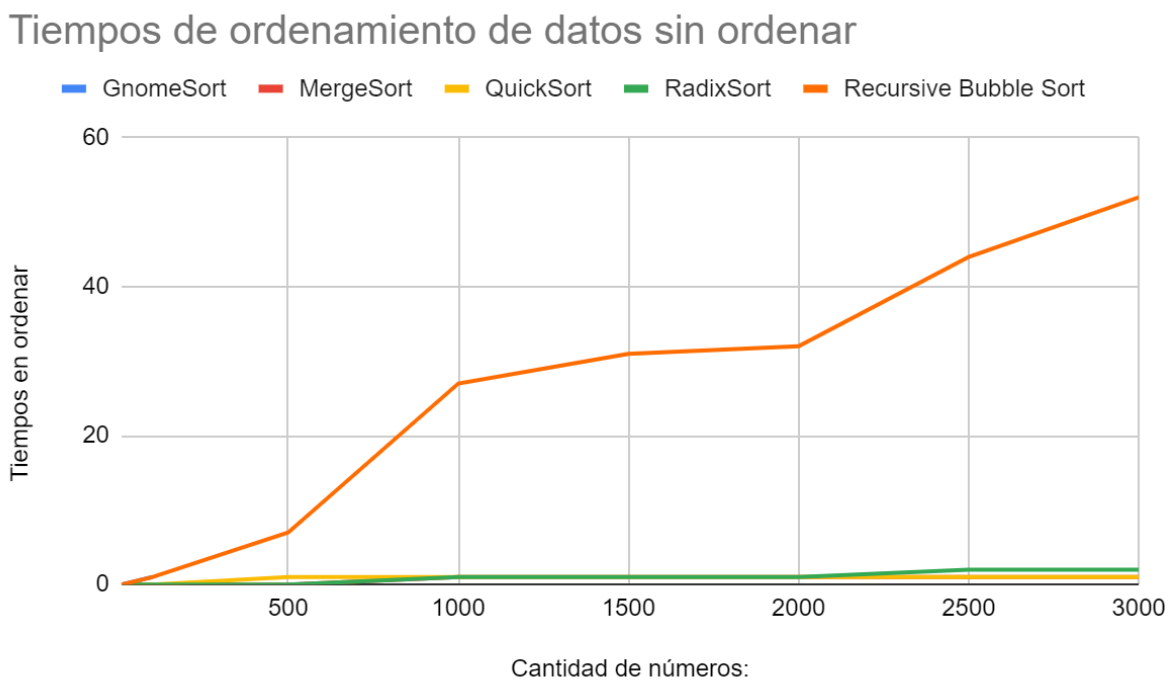
Tablas de resultados de Profiliers:

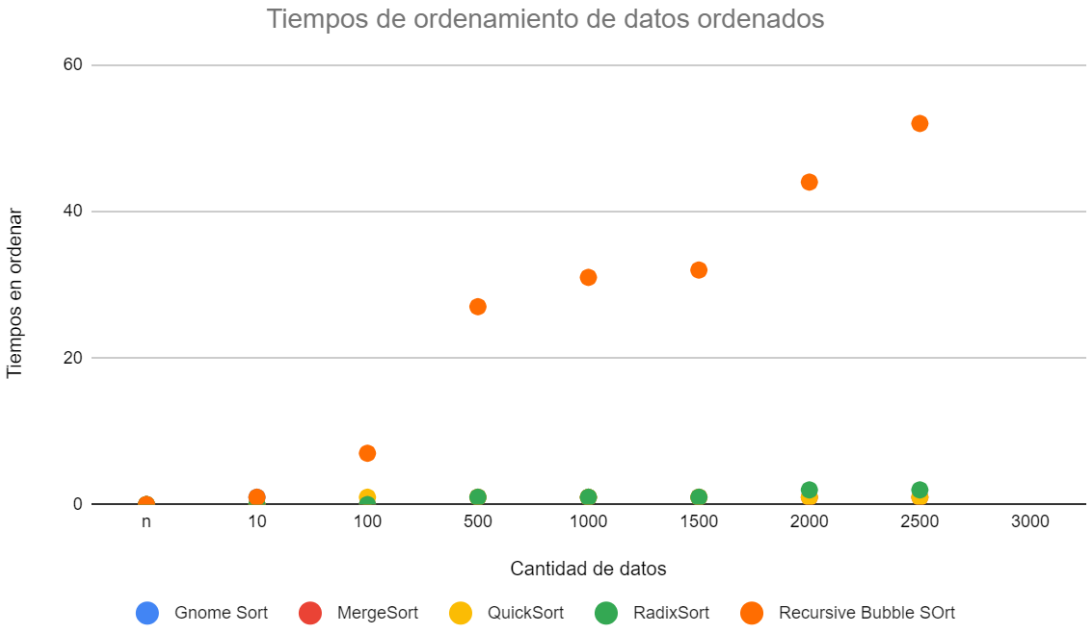
Cantidad de tiempo con números sin ordenar:					
n	**GnomeSort**	**MergeSort**	**QuickSort**	**RadixSort**	**Recursive Bubble Sort**
10	0	0	0	0	0
100	1	0	0	0	1
500		0	1	0	7
1000		1	1	1	27
1500		1	1	1	31
2000		1	1	1	32
2500		1	1	2	44
3000		1	1	2	52
Cantidad de tiempo con números previamente ordenados:					
n	**GnomeSort**	**MergeSort**	**QuickSort**	**RadixSort**	**Recursive Bubble Sort**
10	0	0	0	0	0
100	0	0	0	0	1
500		0	4	0	5
1000		1	11	1	15
1500		1	18	1	22
2000		1	21	1	24

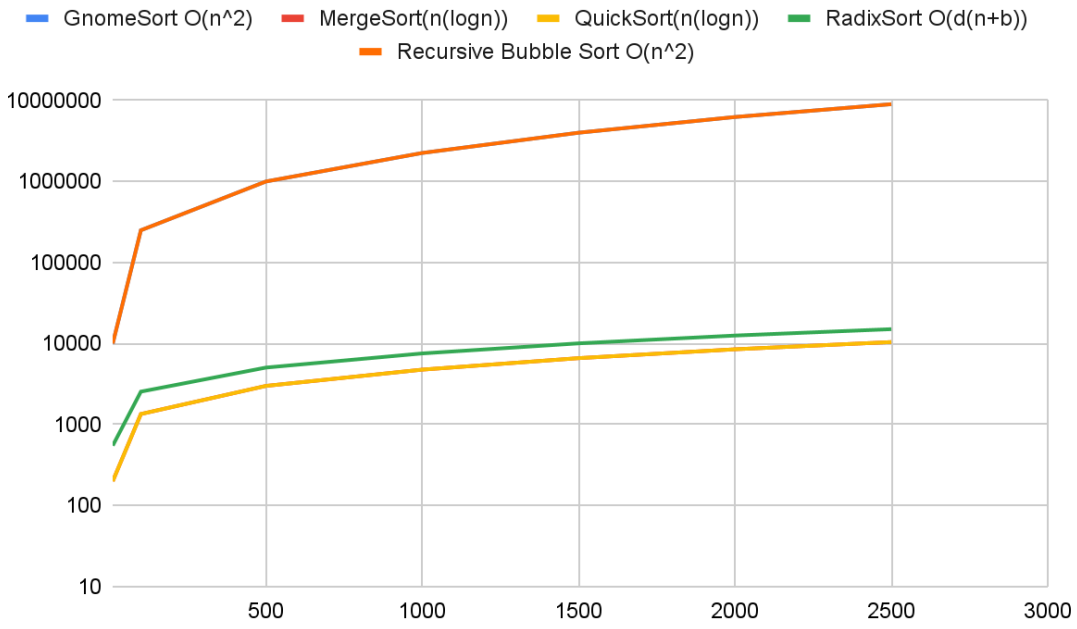
2500		1	31	2	25
3000		1	30	2	26

Complejidad de tiempo big O					
n	**GnomeSort O(n^2)**	**MergeSort(n(logn))**	**QuickSort(n(logn))**	**RadixSort O(d(n+b))**	**Recursive Bubble Sort O(n^2)**
10	100	10,0	10,0	100	100
100	10000	200,0	200,0	550	10000
500	250000	1349,5	1349,5	2550	250000
1000	1000000	3000,0	3000,0	5050	1000000
1500	2250000	4764,1	4764,1	7550	2250000
2000	4000000	6602,1	6602,1	10050	4000000
2500	6250000	8494,9	8494,9	12550	6250000
3000	9000000	10431,4	10431,4	15050	9000000

Gráficos del tiempo que tarda por ordenar cada sort, y su complejidad de tiempo big O:




Tiempos de ordenamiento de datos ordenados
Tiempos en ordenar
60
40
20
0
n
10
100
500
1000
1500
2000
2500
3000
Cantidad de datos
Gnome Sort
MergeSort
QuickSort
RadixSort
Recursive Bubble SOrt


GnomeSort O(n^2)
MergeSort(n(logn))
QuickSort(n(logn))
RadixSort O(d(n+b))
Recursive Bubble Sort O(n^2)
10000000
1000000
100000
10000
1000
100
10
500
1000
1500
2000
2500
3000