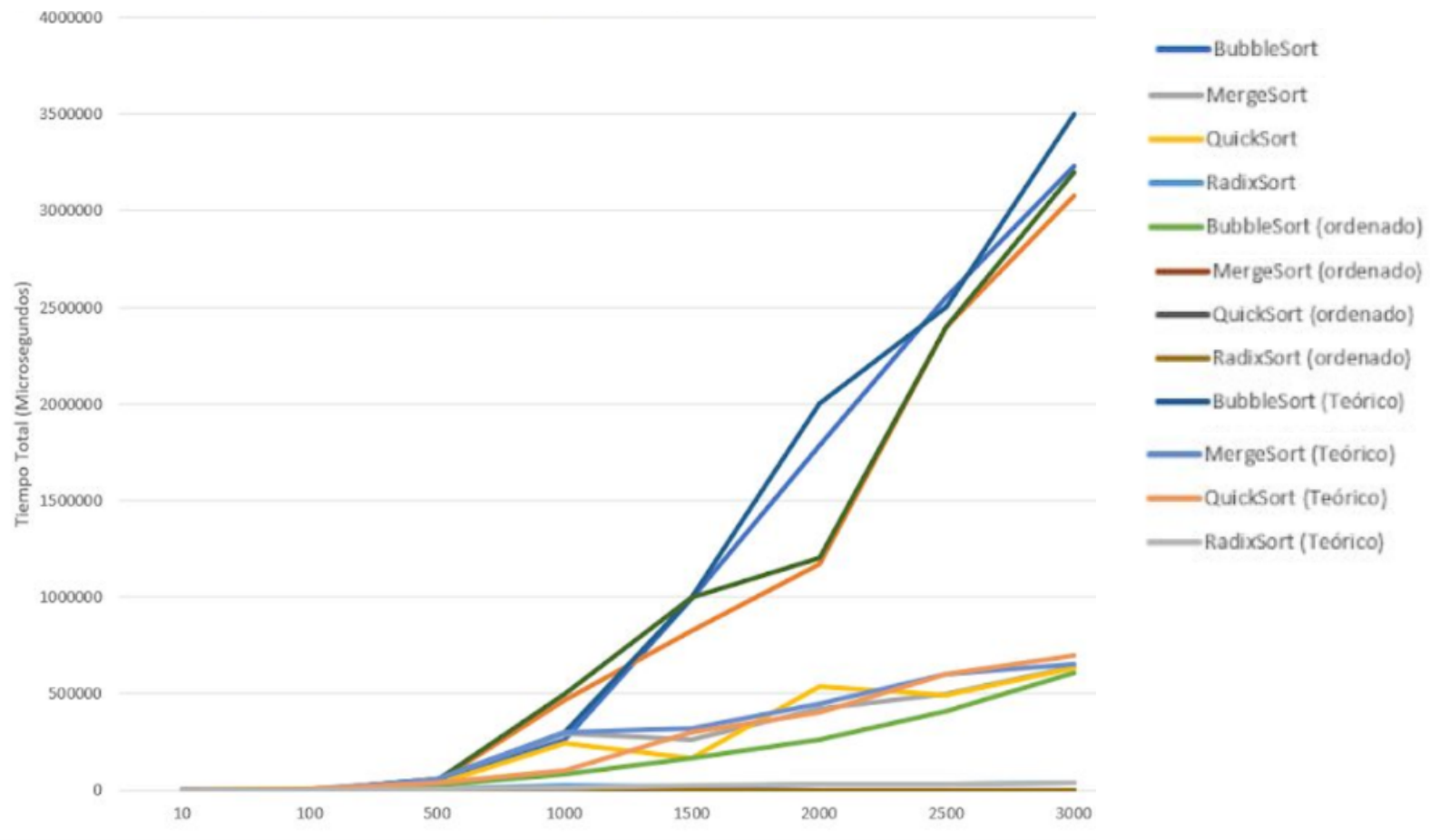


GRAFICA: Tiempo de corrida de los algoritmos-



PROCEDIMIENTO:

Como ya se tenía JProfile en el equipo un fue necesario hacer algún otro tipo de procedimiento para su instalación y configuración. En Eclipse se realizaron las pruebas con los tiempos:

DANIELA VILLAMAR 19086

- 10
- 100
- 500
- 1000
- 15000
- 2000
- 2500
- 3000

Las pruebas en Eclipse se realizaron dos veces.

1. Primera prueba: Arreglo de datos desordenado
2. Segunda Prueba: Arreglo de datos ordenados

Seguidamente se muestran los Screenshots de las pruebas con los diferentes tiempos con los datos que se obtuvieron en base a los métodos del programa que se creó. Por medio de Excel se ordenaron y unieron los datos, además se realizó el diseño de la gráfica del inciso a del tiempo en microsegundos al ejecutar un sort. Para finalizar solo se le agregaron ciertos detalles extras a la gráfica para que pudiera ser más legible.

SCREENSHOTS DATOS DESORDENADOS

-10

The screenshot displays the JProfiler 10.1.5 interface. The left sidebar contains various tool categories: Telemetries, Live memory, Heap Walker, CPU views, Call Tree, Hot Spots, Call Graph, Method Statistics (selected), Complexity Analysis, Call Tracer, JavaScript XHR, Threads, Monitors & locks, Databases, JEE & Probes, and MBeans. The main window shows a table of method statistics for the thread status 'All states'.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.MergeSort.mergesort(java.lang...	3,974 µs	19	209 µs	0 µs	0 µs	3,811 µs	853 µs	3811
application.MergeSort.merge(java.lang.Co...	3,777 µs	9	419 µs	2 µs	2 µs	3,719 µs	1,170 µs	1858.5
application.QuickSort.quickSort(int, int)	162 µs	8	20 µs	4 µs	4 µs	82 µs	26 µs	19.5
application.RadixSort.radixSort(java.lang...	106 µs	1	106 µs	106 µs	106 µs	106 µs	19 µs	0
application.BubbleSort.bubble(java.lang.C...	104 µs	1	104 µs	104 µs	104 µs	104 µs	21 µs	0
application.QuickSort.sort(java.lang.Comp...	88 µs	1	88 µs	88 µs	88 µs	88 µs	13 µs	0
application.RadixSort.countSort(java.lang...	62 µs	1	62 µs	62 µs	62 µs	62 µs	13 µs	0
application.RadixSort.getMax(java.lang.C...	35 µs	1	35 µs	35 µs	35 µs	35 µs	35 µs	0
application.QuickSort.exchangeNumbers(n...	15 µs	12	1 µs	1 µs	1 µs	3 µs	1 µs	2

Below the table is a search bar labeled 'Class View Filters'. The bottom status bar indicates '1 active recording' and 'VM #1 00:17 Profiling'. A watermark 'JProfiler' is visible on the left side of the main window.

The screenshot displays the JProfiler 10.1.5 application window. The left sidebar contains various analysis tools, with 'Method Statistics' selected. The main panel shows a table of method statistics for the thread 'All states'. The table includes columns for Method, Total Time, Inv., Avg. Time, Median Time, Min. Time, Max. Time, Std. Dev., and Outlier Co... The data shows that 'application.MergeSort.mergesort' is the most time-consuming method, followed by 'application.BubbleSort.bubble' and 'application.QuickSort.quickSort'. The status bar at the bottom indicates '1 active recording' and 'VM #1'.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.MergeSort.mergesort(java.lang.C...	23,280 µs	199	116 µs	0 µs	0 µs	4,350 µs	595 µs	4350
application.BubbleSort.bubble(java.lang.C...	3,309 µs	1	3,309 µs	3,309 µs	3,309 µs	3,309 µs	16 µs	0
application.QuickSort.quickSort(int, int)	2,687 µs	85	31 µs	2 µs	2 µs	606 µs	85 µs	302
application.MergeSort.merge(java.lang.Co...	2,300 µs	99	23 µs	1 µs	1 µs	1,797 µs	177 µs	1796
application.QuickSort.sort(java.lang.Comp...	612 µs	1	612 µs	612 µs	612 µs	612 µs	13 µs	0
application.RadixSort.radixsort(java.lang....	358 µs	1	358 µs	358 µs	358 µs	358 µs	17 µs	0
application.RadixSort.countSort(java.lang....	290 µs	2	145 µs	140 µs	140 µs	150 µs	25 µs	0.07
application.QuickSort.exchangeNumbers(in...	123 µs	204	0 µs	0 µs	0 µs	3 µs	0 µs	3
application.RadixSort.getMax(java.lang.C...	59 µs	1	59 µs	59 µs	59 µs	59 µs	16 µs	0

Please select a method above

unlicensed copy for evaluation purposes, 10 days remaining | 1 active recording | VM #1 | 00:18 | Profiling

-500

The screenshot displays the JProfiler 10.1.5 application window. The left sidebar contains various analysis tools, with 'Method Statistics' selected. The main panel shows a table of method statistics for the thread 'All states'. The table includes columns for Method, Total Time, Inv., Avg. Time, Median Time, Min. Time, Max. Time, Std. Dev., and Outlier Co... The data shows that 'application.MergeSort.mergesort' is the most time-consuming method, followed by 'application.BubbleSort.bubble' and 'application.QuickSort.quickSort'. Below the table, there is a search bar for 'Class View Filters' and a message 'Please select a method above'. The bottom status bar indicates '1 active recording' and 'VM #1'.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.MergeSort.mergesort(java.lan...	23,280 µs	199	116 µs	0 µs	0 µs	4,350 µs	595 µs	4350
application.BubbleSort.bubble(java.lang.C...	3,309 µs	1	3,309 µs	3,309 µs	3,309 µs	3,309 µs	16 µs	0
application.QuickSort.quickSort(int, int)	2,687 µs	85	31 µs	2 µs	2 µs	606 µs	85 µs	302
application.MergeSort.merge(java.lang.Co...	2,300 µs	99	23 µs	1 µs	1 µs	1,797 µs	177 µs	1796
application.QuickSort.sort(java.lang.Comp...	612 µs	1	612 µs	612 µs	612 µs	612 µs	13 µs	0
application.RadixSort.radixsort(java.lang....	358 µs	1	358 µs	358 µs	358 µs	358 µs	17 µs	0
application.RadixSort.countSort(java.lang....	290 µs	2	145 µs	140 µs	140 µs	150 µs	25 µs	0.07
application.QuickSort.exchangeNumbers(in...	123 µs	204	0 µs	0 µs	0 µs	3 µs	0 µs	3
application.RadixSort.getMax(java.lang.C...	59 µs	1	59 µs	59 µs	59 µs	59 µs	16 µs	0

DANIELA VILLAMAR 19086

-1000

The screenshot displays the JProfiler 10.1.5 application window. The left sidebar contains various monitoring tools, with 'Method Statistics' selected. The main panel shows a table of method statistics for the thread 'All states'. The table includes columns for Method, Total Time, Inv., Avg. Time, Median Time, Min. Time, Max. Time, Std. Dev., and Outlier Co... The data shows that application.MergeSort.mergesort is the most frequent method, with a total time of 56,244 µs and an average time of 56 µs. Other methods include application.BubbleSort.bubble, application.QuickSort.quickSort, application.MergeSort.merge, application.RadixSort.radixsort, application.RadixSort.countSort, application.QuickSort.exchangeNumbers, and application.RadixSort.getMax.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.MergeSort.mergesort(java.lan...	56,244 µs	999	56 µs	0 µs	0 µs	7,831 µs	396 µs	7831
application.BubbleSort.bubble(java.lang.C...	51,854 µs	1	51,854 µs	51,854 µs	51,854 µs	51,854 µs	21 µs	0
application.QuickSort.quickSort(int, int)	24,841 µs	425	58 µs	1 µs	1 µs	3,691 µs	258 µs	3690
application.QuickSort.sort(java.lang.Comp...	3,697 µs	1	3,697 µs	3,697 µs	3,697 µs	3,697 µs	22 µs	0
application.MergeSort.merge(java.lang.Co...	3,529 µs	499	7 µs	1 µs	1 µs	319 µs	20 µs	318
application.RadixSort.radixsort(java.lang....	1,732 µs	1	1,732 µs	1,732 µs	1,732 µs	1,732 µs	7 µs	0
application.RadixSort.countSort(java.lang....	1,508 µs	3	502 µs	485 µs	485 µs	530 µs	25 µs	0.09
application.QuickSort.exchangeNumbers(in...	669 µs	1,236	0 µs	0 µs	0 µs	11 µs	0 µs	11
application.RadixSort.getMax(java.lang.C...	215 µs	1	215 µs	215 µs	215 µs	215 µs	10 µs	0

Below the table, there is a search bar labeled 'Class View Filters' and a message 'Please select a method above'.

The bottom status bar indicates '1 active recording' and 'VM #1 00:19 Profiling'.

-1500

The screenshot displays the JProfiler 10.1.5 interface. The left sidebar contains various tool categories: Telemetries, Live memory, Heap Walker, CPU views, Call Tree, Hot Spots, Call Graph, Method Statistics (selected), Complexity Analysis, Call Tracer, JavaScript XHR, Threads, Monitors & locks, Databases, JEE & Probes, and MBeans. The main window shows the 'Method Statistics' view for the thread 'All states'. A table lists methods with their performance metrics. The method 'application.MergeSort.mergesort(java.lang.Object[])' is highlighted.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.MergeSort.mergesort(java.lang.Object[])	271 ms	1,999	135 µs	0 µs	0 µs	37,197 µs	1,294 µs	37197
application.BubbleSort.bubble(java.lang.Object[])	263 ms	1	263 ms	263 ms	263 ms	263 ms	12 µs	0
application.QuickSort.quickSort(int, int)	209 ms	867	242 µs	5 µs	5 µs	26,917 µs	1,455 µs	5382.4
application.QuickSort.sort(java.lang.Comparable[], int, int)	26,936 µs	1	26,936 µs	26,936 µs	26,936 µs	26,936 µs	11 µs	0
application.MergeSort.merge(java.lang.Comparable[], int, int)	22,892 µs	999	22 µs	3 µs	3 µs	1,891 µs	99 µs	629.33
application.RadixSort.radixSort(java.lang.Comparable[], int, int)	11,507 µs	1	11,507 µs	11,507 µs	11,507 µs	11,507 µs	18 µs	0
application.RadixSort.countSort(java.lang.Comparable[], int, int)	10,060 µs	3	3,353 µs	3,292 µs	3,292 µs	3,418 µs	62 µs	0.04
application.QuickSort.exchangeNumbers(int[], int, int)	4,706 µs	2,652	1 µs	1 µs	1 µs	34 µs	1 µs	33
application.RadixSort.getMax(java.lang.Comparable[], int, int)	1,407 µs	1	1,407 µs	1,407 µs	1,407 µs	1,407 µs	18 µs	0

Below the table is a search bar labeled 'Q - Class View Filters' and a message 'Please select a method above'.

The status bar at the bottom indicates 'unlicensed copy for evaluation purposes, 10 days remaining', '1 active recording', 'VM #1', '00:28', and 'Profiling'.

DANIELA VILLAMAR 19086

-2000

The screenshot displays the JProfiler 10.1.5 application window. The left sidebar contains various tool categories: Telemetries, Live memory, Heap Walker, CPU views, Call Tree, Hot Spots, Call Graph, Method Statistics (selected), Complexity Analysis, Call Tracer, JavaScript XHR, Threads, Monitors & locks, Databases, JEE & Probes, and MBeans. The main panel shows a table of method statistics for the thread status 'All states'. The table columns are Method, Total Time, Inv., Avg. Time, Median Time, Min. Time, Max. Time, Std. Dev., and Outlier Co... The data rows show performance metrics for various sorting algorithms. The status bar at the bottom indicates '1 active recording' and 'Profiling'.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.BubbleSort.bubble(java.lang.C...	989 ms	1	989 ms	989 ms	989 ms	989 ms	4 µs	0
application.MergeSort.mergeSort(java.lan...	236 ms	2,999	78 µs	0 µs	0 µs	33,034 µs	867 µs	33034
application.QuickSort.quickSort(int, int)	148 ms	1,300	114 µs	2 µs	2 µs	17,811 µs	748 µs	8904.5
application.MergeSort.merge(java.lang.Co...	23,175 µs	1,499	15 µs	1 µs	1 µs	1,691 µs	70 µs	1690
application.QuickSort.sort(java.lang.Comp...	17,819 µs	1	17,819 µs	17,819 µs	17,819 µs	17,819 µs	6 µs	0
application.RadixSort.radixSort(java.lang...	10,997 µs	1	10,997 µs	10,997 µs	10,997 µs	10,997 µs	22 µs	0
application.RadixSort.countSort(java.lang...	9,672 µs	4	2,418 µs	2,175 µs	1,967 µs	3,224 µs	478 µs	0.48
application.QuickSort.exchangeNumbers(n...	3,046 µs	4,192	0 µs	0 µs	0 µs	13 µs	0 µs	13
application.RadixSort.getMax(java.lang.C...	1,295 µs	1	1,295 µs	1,295 µs	1,295 µs	1,295 µs	20 µs	0

Please select a method above

unlicensed copy for evaluation purposes, 10 days remaining | 1 active recording | VM #1 | 00:22 | Profiling

-2500

The screenshot displays the JProfiler 10.1.5 application window. The left sidebar contains various analysis tools, with 'Method Statistics' selected. The main panel shows a table of method statistics for the thread 'All states'. The table includes columns for Method, Total Time, Inv., Avg. Time, Median Time, Min. Time, Max. Time, Std. Dev., and Outlier Coefficient. The first method listed is 'application.BubbleSort.bubble(java.lang.C...)' with a total time of 1,781 ms. Below the table is a search bar for 'Class View Filters' and a message 'Please select a method above'. The bottom status bar indicates '1 active recording' and 'VM #1'.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.BubbleSort.bubble(java.lang.C...	1,781 ms	1	1,781 ms	1,781 ms	1,781 ms	1,781 ms	0 µs	0
application.QuickSort.quickSort(int, int)	470 ms	1,727	272 µs	2 µs	2 µs	55,363 µs	2,117 µs	27680.5
application.MergeSort.mergesort(java.lan...	382 ms	3,999	95 µs	0 µs	0 µs	51,483 µs	1,145 µs	51483
application.QuickSort.sort(java.lang.Comp...	55,384 µs	1	55,384 µs	55,384 µs	55,384 µs	55,384 µs	9 µs	0
application.MergeSort.merge(java.lang.Co...	37,876 µs	1,999	18 µs	2 µs	2 µs	2,923 µs	95 µs	1460.5
application.RadixSort.radixsort(java.lang....	17,225 µs	1	17,225 µs	17,225 µs	17,225 µs	17,225 µs	0 µs	0
application.RadixSort.countSort(java.lang....	15,235 µs	4	3,808 µs	3,733 µs	3,733 µs	3,898 µs	75 µs	0.04
application.QuickSort.exchangeNumbers(in...	9,955 µs	5,882	1 µs	0 µs	0 µs	77 µs	1 µs	77
application.RadixSort.getMax(java.lang.C...	1,960 µs	1	1,960 µs	1,960 µs	1,960 µs	1,960 µs	15 µs	0

DANIELA VILLAMAR 19086

-3000

The screenshot displays the JProfiler 10.1.5 application window. The left sidebar contains various analysis tools, with 'Method Statistics' selected. The main panel shows a table of method statistics for the thread 'All states'. The table includes columns for Method, Total Time, Inv., Avg. Time, Median Time, Min. Time, Max. Time, Std. Dev., and Outlier Co... The first row shows 'application.BubbleSort.bubble(java.lang.C...' with a total time of 2,554 ms and 1 invocation. The second row shows 'application.MergeSort.mergeSort(java.lan...' with a total time of 446 ms and 4,999 invocations. The third row shows 'application.QuickSort.quickSort(int, int)' with a total time of 433 ms and 2,151 invocations. The fourth row shows 'application.MergeSort.merge(java.lang.Co...' with a total time of 53,930 ms and 2,499 invocations. The fifth row shows 'application.QuickSort.sort(java.lang.Comp...' with a total time of 53,181 ms and 1 invocation. The sixth row shows 'application.RadixSort.radixsort(java.lang...' with a total time of 15,155 ms and 1 invocation. The seventh row shows 'application.RadixSort.countSort(java.lang...' with a total time of 13,483 ms and 4 invocations. The eighth row shows 'application.QuickSort.exchangeNumbers(in...' with a total time of 8,982 ms and 7,473 invocations. The ninth row shows 'application.RadixSort.getMax(java.lang.C...' with a total time of 1,643 ms and 1 invocation. The bottom status bar indicates 'unlicensed copy for evaluation purposes, 10 days remaining', '3 overhead hot spots', '1 active recording', 'VM #1', '01:33', and 'Profiling'.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.BubbleSort.bubble(java.lang.C...	2,554 ms	1	2,554 ms	2,554 ms	2,554 ms	2,554 ms	10 μs	0
application.MergeSort.mergeSort(java.lan...	446 ms	4,999	89 μs	0 μs	0 μs	64,468 μs	1,249 μs	64468
application.QuickSort.quickSort(int, int)	433 ms	2,151	201 μs	2 μs	2 μs	53,159 μs	1,676 μs	26578.5
application.MergeSort.merge(java.lang.Co...	53,930 ms	2,499	21 μs	2 μs	2 μs	6,112 μs	176 μs	3055
application.QuickSort.sort(java.lang.Comp...	53,181 μs	1	53,181 μs	53,181 μs	53,181 μs	53,181 μs	6 μs	0
application.RadixSort.radixsort(java.lang...	15,155 μs	1	15,155 μs	15,155 μs	15,155 μs	15,155 μs	20 μs	0
application.RadixSort.countSort(java.lang...	13,483 μs	4	3,370 μs	2,625 μs	2,457 μs	4,369 μs	846 μs	0.66
application.QuickSort.exchangeNumbers(in...	8,982 μs	7,473	1 μs	0 μs	0 μs	44 μs	1 μs	44
application.RadixSort.getMax(java.lang.C...	1,643 μs	1	1,643 μs	1,643 μs	1,643 μs	1,643 μs	18 μs	0

DANIELA VILLAMAR 19086

SCREENSHOTS DATOS ORDENADOS

-10

Thread status: All states

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.RadixSort.radixsort(java.lang....	3,798 µs	1	3,798 µs	3,798 µs	3,798 µs	3,798 µs	23 µs	0
application.RadixSort.countSort(java.lang....	3,762 µs	1	3,762 µs	3,762 µs	3,762 µs	3,762 µs	13 µs	0
application.MergeSort.mergeSort(java.lan...	248 µs	19	13 µs	0 µs	0 µs	103 µs	31 µs	103
application.QuickSort.quickSort(int, int)	120 µs	6	20 µs	5 µs	5 µs	74 µs	28 µs	13.8
application.QuickSort.sort(java.lang.Comp...	81 µs	1	81 µs	81 µs	81 µs	81 µs	6 µs	0
application.BubbleSort.bubble(java.lang.Co...	72 µs	1	72 µs	72 µs	72 µs	72 µs	3 µs	0
application.MergeSort.merge(java.lang.Co...	69 µs	9	7 µs	2 µs	2 µs	19 µs	7 µs	8.5
application.RadixSort.getMax(java.lang.Co...	28 µs	1	28 µs	28 µs	28 µs	28 µs	28 µs	0
application.QuickSort.exchangeNumbers(n...	10 µs	6	1 µs	0 µs	0 µs	4 µs	1 µs	4

Please select a method above

unlicensed copy for evaluation purposes, 10 days remaining

1 active recording

VM #1 00:16 Profiling

The screenshot displays the JProfiler 10.1.5 application window. The left sidebar contains various tool categories: Telemetries, Live memory, Heap Walker, CPU views, Call Tree, Hot Spots, Call Graph, Method Statistics (selected), Complexity Analysis, Call Tracer, JavaScript XHR, Threads, Monitors & locks, Databases, JEE & Probes, and MBeans. The main panel shows a table of method statistics for the selected thread status 'All states'.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.MergeSort.mergeSort(java.lang.Co...	24,384 µs	199	122 µs	0 µs	0 µs	4,532 µs	616 µs	4532
application.MergeSort.merge(java.lang.Co...	2,414 µs	99	24 µs	1 µs	1 µs	1,868 µs	187 µs	1867
application.BubbleSort.bubble(java.lang.C...	2,057 µs	1	2,057 µs	2,057 µs	2,057 µs	2,057 µs	18 µs	0
application.QuickSort.quickSort(int, int)	1,704 µs	63	27 µs	2 µs	2 µs	458 µs	70 µs	228
application.QuickSort.sort(java.lang.Comp...	464 µs	1	464 µs	464 µs	464 µs	464 µs	11 µs	0
application.RadixSort.radixsort(java.lang...	389 µs	1	389 µs	389 µs	389 µs	389 µs	14 µs	0
application.RadixSort.countSort(java.lang...	293 µs	2	146 µs	125 µs	118 µs	175 µs	25 µs	0.4
application.RadixSort.getMax(java.lang.C...	86 µs	1	86 µs	86 µs	86 µs	86 µs	11 µs	0
application.QuickSort.exchangeNumbers(n...	44 µs	63	0 µs	0 µs	0 µs	3 µs	0 µs	3

Below the table is a search bar labeled 'Class View Filters' and a message 'Please select a method above'. The bottom status bar indicates '1 active recording', 'VM #1', '00:28', and 'Profiling'.

-500

Thread status: All states

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.MergeSort.mergesort(java.lan...	56,699 µs	999	56 µs	0 µs	0 µs	7,996 µs	402 µs	7996
application.BubbleSort.bubble(java.lang.C...	26,214 µs	1	26,214 µs	26,214 µs	26,214 µs	26,214 µs	11 µs	0
application.QuickSort.quickSort(int, int)	11,984 µs	255	46 µs	2 µs	2 µs	2,427 µs	190 µs	1212.5
application.MergeSort.merge(java.lang.Co...	3,600 µs	499	7 µs	1 µs	1 µs	551 µs	29 µs	550
application.QuickSort.sort(java.lang.Comp...	2,435 µs	1	2,435 µs	2,435 µs	2,435 µs	2,435 µs	10 µs	0
application.RadixSort.radixsort(java.lang....	2,094 µs	1	2,094 µs	2,094 µs	2,094 µs	2,094 µs	19 µs	0
application.RadixSort.countSort(java.lang....	1,741 µs	3	580 µs	575 µs	557 µs	624 µs	26 µs	0.09
application.RadixSort.getMax(java.lang.C...	343 µs	1	343 µs	343 µs	343 µs	343 µs	18 µs	0
application.QuickSort.exchangeNumbers(in...	177 µs	255	0 µs	0 µs	0 µs	5 µs	0 µs	5

Please select a method above

unlicensed copy for evaluation purposes, 10 days remaining

1 active recording

VM #1 00:24 Profiling

DANIELA VILLAMAR 19086

-1000

Main (9) - JProfiler 10.1.5

Session View Profiling Window Help

Start Center Activate IDE Save Snapshot Session Settings Start Recordings Stop Recordings Start Tracking Run GC Add Bookmark Export View Settings Help Record Statistics

Thread status: All states

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co.
application.BubbleSort.bubble(java.lang.C...	81,187 µs	1	81,187 µs	81,187 µs	81,187 µs	81,187 µs	12 µs	0
application.MergeSort.mergesort(java.lan...	80,686 µs	1,999	40 µs	0 µs	0 µs	10,537 µs	356 µs	10537
application.QuickSort.quickSort(int, int)	45,680 µs	511	89 µs	2 µs	2 µs	7,868 µs	461 µs	3933
application.QuickSort.sort(java.lang.Comp...	7,878 µs	1	7,878 µs	7,878 µs	7,878 µs	7,878 µs	3 µs	0
application.RadixSort.radixsort(java.lang...	7,447 µs	1	7,447 µs	7,447 µs	7,447 µs	7,447 µs	22 µs	0
application.MergeSort.merge(java.lang.Co...	6,240 µs	999	6 µs	0 µs	0 µs	386 µs	21 µs	386
application.RadixSort.countSort(java.lang...	6,193 µs	3	2,064 µs	2,025 µs	2,002 µs	2,160 µs	71 µs	0.07
application.RadixSort.getMax(java.lang.C...	1,237 µs	1	1,237 µs	1,237 µs	1,237 µs	1,237 µs	12 µs	0
application.QuickSort.exchangeNumbers(in...	584 µs	511	1 µs	0 µs	0 µs	20 µs	1 µs	20

Please select a method above

unlicensed copy for evaluation purposes, 10 days remaining

1 active recording VM #1 00:19 Profiling

-1500

The screenshot displays the JProfiler 10.1.5 application window. The left sidebar contains various tool categories: Telemetries, Live memory, Heap Walker, CPU views, Call Tree, Hot Spots, Call Graph, Method Statistics (selected), Complexity Analysis, Call Tracer, JavaScript XHR, Threads, Monitors & locks, Databases, JEE & Probes, and MBeans. The main panel shows a table of Method Statistics for the selected method, application.BubbleSort.bubble. The table includes columns for Method, Total Time, Inv., Avg. Time, Median Time, Min. Time, Max. Time, Std. Dev., and Outlier Co... The status bar at the bottom indicates '1 active recording' and 'VM #1 00:14 Profiling'.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.BubbleSort.bubble(java.lang.C...	158 ms	1	158 ms	158 ms	158 ms	158 ms	1 μs	0
application.MergeSort.mergeSort(java.lan...	101 ms	2,999	33 μs	0 μs	0 μs	14,700 μs	392 μs	14700
application.QuickSort.quickSort(int, int)	92,400 μs	988	93 μs	3 μs	3 μs	15,311 μs	628 μs	5102.67
application.QuickSort.sort(java.lang.Comp...	15,326 μs	1	15,326 μs	15,326 μs	15,326 μs	15,326 μs	1 μs	0
application.RadixSort.radixsort(java.lang....	11,465 μs	1	11,465 μs	11,465 μs	11,465 μs	11,465 μs	10 μs	0
application.MergeSort.merge(java.lang.Co...	10,150 μs	1,499	6 μs	0 μs	0 μs	1,230 μs	41 μs	1230
application.RadixSort.countSort(java.lang.C...	9,620 μs	4	2,405 μs	2,025 μs	1,481 μs	3,107 μs	691 μs	0.53
application.RadixSort.getMax(java.lang.C...	1,825 μs	1	1,825 μs	1,825 μs	1,825 μs	1,825 μs	0 μs	0
application.QuickSort.exchangeNumbers(n...	1,162 μs	988	1 μs	0 μs	0 μs	20 μs	1 μs	20

Please select a method above

unlicensed copy for evaluation purposes, 10 days remaining | 1 active recording | VM #1 00:14 | Profiling

-2000

The screenshot displays the JProfiler 10.1.5 application window. The left sidebar contains various analysis tools, with 'Method Statistics' selected. The main panel shows a table of method statistics for the thread 'All states'. The table includes columns for Method, Total Time, Inv., Avg. Time, Median Time, Min. Time, Max. Time, Std. Dev., and Outlier Co... The data is sorted by Total Time, with 'application.BubbleSort.bubble' at the top.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.BubbleSort.bubble(java.lang.C...	260 ms	1	260 ms	260 ms	260 ms	260 ms	17 µs	0
application.MergeSort.mergesort(java.lan...	123 ms	3,999	30 µs	0 µs	0 µs	15,592 µs	365 µs	15592
application.QuickSort.quickSort(int, int)	56,297 µs	1,023	55 µs	2 µs	2 µs	9,377 µs	378 µs	4687.5
application.MergeSort.merge(java.lang.Co...	10,676 µs	1,999	5 µs	0 µs	0 µs	748 µs	23 µs	748
application.QuickSort.sort(java.lang.Comp...	9,383 µs	1	9,383 µs	9,383 µs	9,383 µs	9,383 µs	8 µs	0
application.RadixSort.radixsort(java.lang....	9,080 µs	1	9,080 µs	9,080 µs	9,080 µs	9,080 µs	5 µs	0
application.RadixSort.countSort(java.lang....	7,797 µs	4	1,949 µs	1,925 µs	1,909 µs	2,001 µs	43 µs	0.04
application.RadixSort.getMax(java.lang.C...	1,271 µs	1	1,271 µs	1,271 µs	1,271 µs	1,271 µs	4 µs	0
application.QuickSort.exchangeNumbers(in...	575 µs	1,023	0 µs	0 µs	0 µs	10 µs	0 µs	10

Below the table, there is a search bar labeled 'Class View Filters' and a message 'Please select a method above'.

The status bar at the bottom indicates '1 active recording' and 'VM #1 00:16 Profiling'.

DANIELA VILLAMAR 19086

-2500

The screenshot displays the JProfiler 10.1.5 application window. The interface includes a menu bar (Session, View, Profiling, Window, Help), a toolbar with icons for session management, profiling, and analysis, and a left sidebar with various tool categories. The main panel shows a table of method statistics for the 'Thread status: All states'.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.BubbleSort.bubble(java.lang.C...	411 ms	1	411 ms	411 ms	411 ms	411 ms	7 µs	0
application.MergeSort.mergesort(java.lan...	272 ms	4,999	54 µs	0 µs	0 µs	35,572 µs	759 µs	35572
application.QuickSort.quickSort(int, int)	130 ms	1,476	88 µs	3 µs	3 µs	20,419 µs	693 µs	6805.33
application.MergeSort.merge(java.lang.Co...	28,223 µs	2,499	11 µs	0 µs	0 µs	3,427 µs	87 µs	3427
application.QuickSort.sort(java.lang.Comp...	20,436 µs	1	20,436 µs	20,436 µs	20,436 µs	20,436 µs	11 µs	0
application.RadixSort.radixsort(java.lang....	14,680 µs	1	14,680 µs	14,680 µs	14,680 µs	14,680 µs	5 µs	0
application.RadixSort.countSort(java.lang....	13,250 µs	4	3,312 µs	2,675 µs	2,604 µs	3,991 µs	662 µs	0.49
application.RadixSort.getMax(java.lang.C...	1,411 µs	1	1,411 µs	1,411 µs	1,411 µs	1,411 µs	14 µs	0
application.QuickSort.exchangeNumbers(n...	1,402 µs	1,476	0 µs	0 µs	0 µs	21 µs	0 µs	21

Below the table is a search bar labeled 'Class View Filters'. The status bar at the bottom indicates '1 active recording' and 'VM #1'.

DANIELA VILLAMAR 19086

-3000

The screenshot displays the JProfiler 10.1.5 application window. The main panel shows a table of method statistics for the thread status 'All states'. The table includes columns for Method, Total Time, Inv., Avg. Time, Median Time, Min. Time, Max. Time, Std. Dev., and Outlier Co... The methods listed are application.BubbleSort.bubble, application.MergeSort.mergeSort, application.QuickSort.quickSort, application.MergeSort.merge, application.QuickSort.sort, application.RadixSort.radixsort, application.RadixSort.countSort, application.RadixSort.getMax, and application.QuickSort.exchangeNumbers. The status bar at the bottom indicates '1 active recording' and 'Profiling'.

Method	Total Time	Inv.	Avg. Time	Median Time	Min. Time	Max. Time	Std. Dev.	Outlier Co...
application.BubbleSort.bubble(java.lang.C...	606 ms	1	606 ms	606 ms	606 ms	606 ms	606 ms	13 µs
application.MergeSort.mergeSort(java.lan...	212 ms	5,999	35 µs	0 µs	0 µs	28,208 µs	499 µs	28208
application.QuickSort.quickSort(int, int)	107 ms	1,976	54 µs	1 µs	1 µs	15,967 µs	470 µs	15966
application.MergeSort.merge(java.lang.Co...	21,911 µs	2,999	7 µs	0 µs	0 µs	2,072 µs	49 µs	2072
application.QuickSort.sort(java.lang.Comp...	15,975 µs	1	15,975 µs	15,975 µs	15,975 µs	15,975 µs	0 µs	0
application.RadixSort.radixsort(java.lang....	14,290 µs	1	14,290 µs	14,290 µs	14,290 µs	14,290 µs	15 µs	0
application.RadixSort.countSort(java.lang....	12,268 µs	4	3,067 µs	3,125 µs	2,833 µs	3,164 µs	145 µs	0.01
application.RadixSort.getMax(java.lang.C...	1,998 µs	1	1,998 µs	1,998 µs	1,998 µs	1,998 µs	23 µs	0
application.QuickSort.exchangeNumbers(n...	1,223 µs	1,976	0 µs	0 µs	0 µs	19 µs	0 µs	19

Please select a method above