

# Assignment6

## screenshot

The screenshot of mergesort test.

The screenshot shows an IDE interface with the following details:

- Title Bar:** Github – MergeSortTest.java
- Project Explorer:** Shows a package structure with several sorting algorithms and their corresponding test classes (e.g., BubbleSort, BubbleSortTest, HeapSort, HeapSortTest, InsertionSort, InsertionSortBasic, InsertionSortMSD, InsertionSortMSDT, InsertionSortOpt, InsertionSortOptTest, RandomSort, RandomSortTest).
- Code Editor:** The current file is MergeSortTest.java, which contains Java code for a merge sort test. It includes imports, a class definition, and a beforeClass annotation.
- Run Tab:** Shows a successful run of the MergeSortTest class with 15 tests passed in 668ms.
- Output Tab:** Displays the execution logs for the tests, including instrumentation details for insertion sort and merge sort helpers.
- Bottom Status Bar:** Shows the time (24:14), file type (LF), encoding (UTF-8), spaces (4 spaces), and the Spring2023 semester.

For each experiment (a sort method of a given size), you will run it twice: once for the instrumentation and the timing under `MergeSortTest`, `QuickSortDualPivotTest` and `HeapSortTest.java`.

- HeapSort Timing

```
156     final int compares = (int) getStatPack.getStatistics(InstrumentedHelper.  
157         final int hits = (int) getStatPack.getStatistics(InstrumentedHelper.  
158  
159         System.out.println("Compares: " + compares + "Copies: " + copies + "Hits: " + hits);  
160         System.out.println("~~~~~");  
161     }  
  
HeapSortTest ✘  
» ✓ Tests passed: 7 of 7 tests – 3 min 44 sec  
  
Timing :  
2023-03-12 21:35:18 INFO Benchmark_Timer - Begin run: HeapSort with 100 runs  
N: 10000 -> 1.43ms  
2023-03-12 21:35:18 INFO Benchmark_Timer - Begin run: HeapSort with 100 runs  
N: 20000 -> 3.18ms  
2023-03-12 21:35:18 INFO Benchmark_Timer - Begin run: HeapSort with 100 runs  
N: 40000 -> 6.94ms  
2023-03-12 21:35:19 INFO Benchmark_Timer - Begin run: HeapSort with 100 runs  
N: 80000 -> 15.26ms  
2023-03-12 21:35:21 INFO Benchmark_Timer - Begin run: HeapSort with 100 runs  
N: 128000 -> 26.08ms  
2023-03-12 21:35:23 INFO Benchmark_Timer - Begin run: HeapSort with 100 runs  
N: 256000 -> 62.34ms
```

- HeapSort instrument:

```

final int copies = (int) getStatPack.getStatistics(InstrumentedHelper.COPIES).mean();
final int hits = (int) getStatPack.getStatistics(InstrumentedHelper.HITS).mean();

System.out.println("Compares: " + compares + "Copies: " + copies + "Swaps: " + swaps + "Inversions: " + inversions);
System.out.println("Instrumentation :");

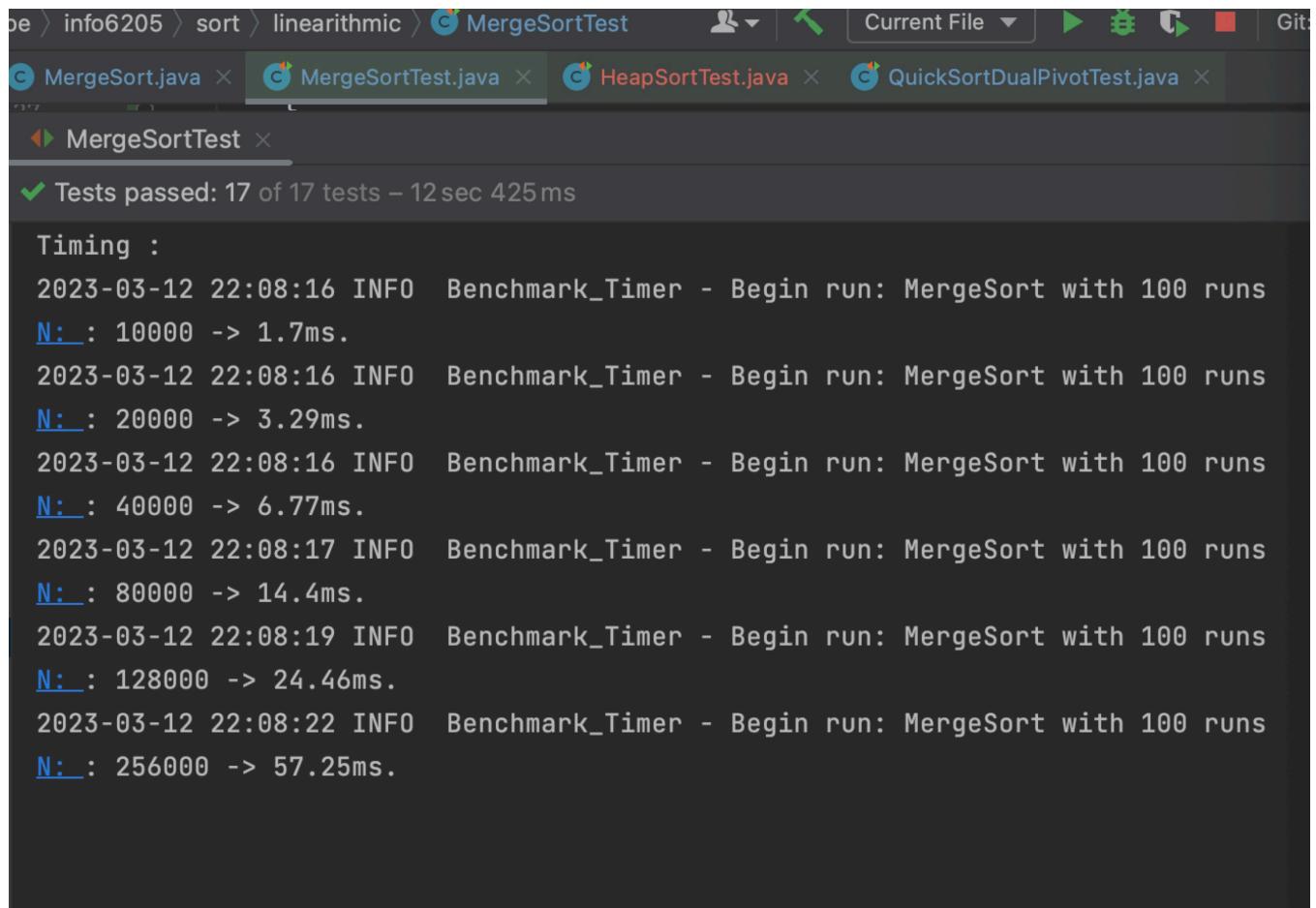
```

Tests passed: 7 of 7 tests – 3 min 44 sec

Instrumentation :

N	Compares	Copies	Swaps	Inversions
10000	235433	0	124187	24744310
20000	510654	0	268324	100053337
40000	1101349	0	576684	400897786
80000	2363211	0	1233857	1602388415
128000	554334048	0	9661850	554334048

- MergeSort Timing



```
be > info6205 > sort > linearithmic > MergeSortTest
MergeSort.java × MergeSortTest.java × HeapSortTest.java × QuickSortDualPivotTest.java ×
MergeSortTest ×
Tests passed: 17 of 17 tests – 12 sec 425 ms

Timing :
2023-03-12 22:08:16 INFO Benchmark_Timer - Begin run: MergeSort with 100 runs
N: 10000 -> 1.7ms.
2023-03-12 22:08:16 INFO Benchmark_Timer - Begin run: MergeSort with 100 runs
N: 20000 -> 3.29ms.
2023-03-12 22:08:16 INFO Benchmark_Timer - Begin run: MergeSort with 100 runs
N: 40000 -> 6.77ms.
2023-03-12 22:08:17 INFO Benchmark_Timer - Begin run: MergeSort with 100 runs
N: 80000 -> 14.4ms.
2023-03-12 22:08:19 INFO Benchmark_Timer - Begin run: MergeSort with 100 runs
N: 128000 -> 24.46ms.
2023-03-12 22:08:22 INFO Benchmark_Timer - Begin run: MergeSort with 100 runs
N: 256000 -> 57.25ms.
```

- MergeSort instrument

GitHub – MergeSortTest.java

Project GitHub src test java edu neu coe info6205 sort linearithmic MergeSortTest Current File Run: QuickSortDualPivotTest MergeSortTest Tests passed: 17 of 17 tests – 12 sec 425 ms

Instrumentation:

N :10000  
Compares: 123495Copies: 109902Swaps: 9758Inversions : 25084627Fixes: 25084627Hits: 273728

N :20000  
Compares: 267117Copies: 239885Swaps: 19577Inversions : 100177821Fixes: 100177821Hits: 587704

N :40000  
Compares: 573902Copies: 519748Swaps: 38929Inversions : 401871722Fixes: 401871722Hits: 1254848

N :80000  
Compares: 1228308Copies: 1119594Swaps: 78228Inversions : 1599615433Fixes: 1599615433Hits: 2670556

N :128000  
Compares: 2052608Copies: 1896993Swaps: 111721Inversions : -204859682Fixes: -204859682Hits: 4429718

N :256000  
Compares: 4361247Copies: 4050194Swaps: 224699Inversions : -808701701Fixes: -808701701Hits: 9375980

- QuickSort Timing

```
Timing;
2023-03-12 22:27:44 INFO Benchmark_Timer - Begin run: QuickSort Dual Pivots
:N 10000 -> 0.0ms
2023-03-12 22:27:44 INFO Benchmark_Timer - Begin run: QuickSort Dual Pivots
:N 20000 -> 0.0ms
2023-03-12 22:27:44 INFO Benchmark_Timer - Begin run: QuickSort Dual Pivots
:N 40000 -> 0.0ms
2023-03-12 22:27:44 INFO Benchmark_Timer - Begin run: QuickSort Dual Pivots
:N 80000 -> 0.0ms
2023-03-12 22:27:44 INFO Benchmark_Timer - Begin run: QuickSort Dual Pivots
:N 128000 -> 0.0ms
2023-03-12 22:27:44 INFO Benchmark_Timer - Begin run: QuickSort Dual Pivots
:N 256000 -> 0.0ms
```

- QuickSort instrument

The screenshot shows a Java IDE interface with the following details:

- Project:** Github - QuickSortDualPivotTest.java
- Run:** QuickSortDualPivotTest Test Results
- Test Summary:** Tests passed: 16 of 16 tests - 1min 52sec
- Test Details:** The results are grouped by array size:
  - 10,000 elements: 151787 Copies, 0 Swaps, 62014 Inversions, 24835299 Fixes, 26745524 Hits, 402765 ms.
  - 20,000 elements: 338489 Copies, 0 Swaps, 143257 Inversions, 99832908 Fixes, 116888905 Hits, 917149 ms.
  - 40,000 elements: 712249 Copies, 0 Swaps, 312090 Inversions, 399394145 Fixes, 412346524 Hits, 1971360 ms.
  - 80,000 elements: 1534839 Copies, 0 Swaps, 626596 Inversions, 1600340564 Fixes, 1657079617 Hits, 4063460 ms.
  - 160,000 elements: 3265825 Copies, 0 Swaps, 1377444 Inversions, 2086294638 Fixes, -2019215870 Hits, 8819483 ms.
  - 256,000 elements: 712249 Copies, 0 Swaps, 312090 Inversions, 399394145 Fixes, 412346524 Hits, 1971360 ms.
- Bottom Status:** Tests passed: 16 (a minute ago)

## Graph & conclusion

HeapSort		time	comparisons	copies	swaps	inversions	fixes	hits
N								
10000	1.42	235334		0	124155	25008297	75655881	966808
20000	3.1	510984		0	268616	99614148	303623413	2095048
40000	6.77	1101322		0	576704	398266242	1.209E+09	4509452
80000	14.79	2363008		0	1233709	1.605E+09	1.855E+09	9660044
160000	33.72	5046625		0	2627510	2.097E+09	2.111E+09	20600986
256000	57.74	8410576		0	4372009	2.897E+09	2.294E+09	34310038

QuickSort Dual Pivots								
N	time	comparisons	copies	swaps	inversions	fixes	hits	
10000	1.35	151787		62014	24835299	26745524	402765	
20000	2.14	338489		143257	99832908	116888905	917149	
40000	5.01	712249		312090	399394145	412346524	1971360	
80000	9.92	1534839		626596	1.6E+09	1.657E+09	4063460	
160000	20.67	3265825		1377444	2.086E+09	2.019E+09	8819483	
256000	33.51	5525002		2190632	2.878E+09	2.367E+09	14357631	

MergeSort								
N	time	comparisons	copies	swaps	inversions	fixes	hits	
10000	1.51	123551	109952		9756	24977728	24977728	273506
20000	2.65	267137	239904		19700	100301422	100301422	588108
40000	5.48	574123	519843		39076	400388388	400388388	1254976
80000	11.69	1228303	1119683		77917	1.599E+09	1.599E+09	2671284
160000	27.6	2616355	2399350		155942	2.105E+09	2.105E+09	5663610
256000	47.61	4361293	4049954		224277	2.905E+09	2.905E+09	9374866

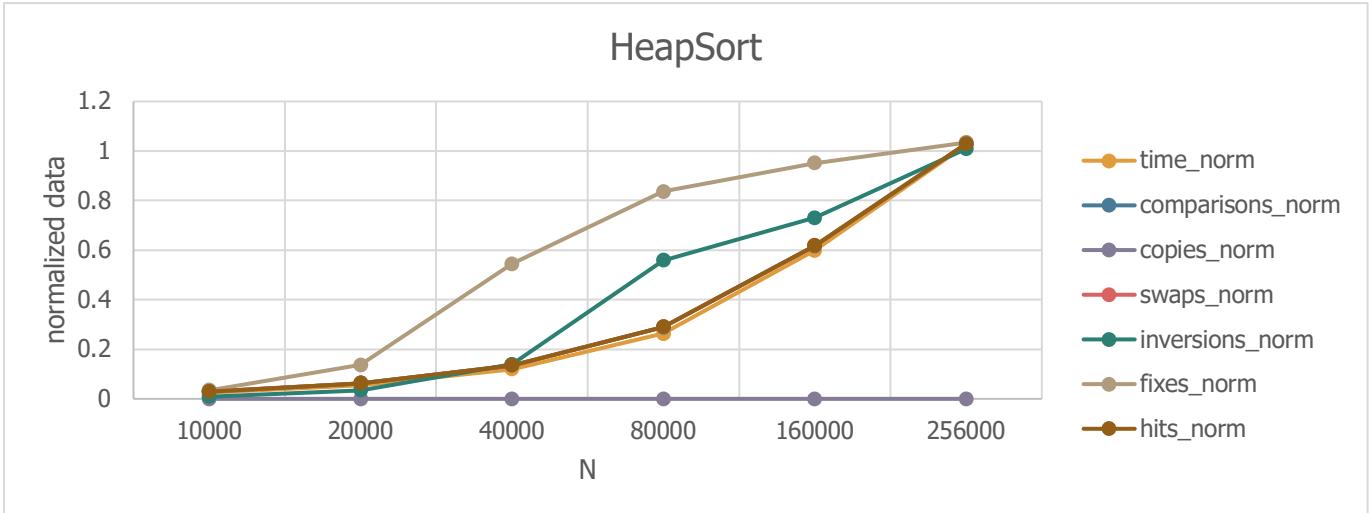
HeapSort														
N	time	comparisons	copies	swaps	inversions	fixes	hits	time_norm	comparisons_norm	copies_norm	swaps_norm	inversions_norm	fixes_norm	hits_norm
10000	1.42	235334	0	124155	25008297	75655881	966808	0.02521307	0.02878618	0	0.0292277	0.00870666	0.03411071	0.02899563
20000	3.1	510984	0	268616	99614148	303623413	2095048	0.0504261	0.06250384	0	0.0632357	0.03468076	0.13689365	0.06283278
40000	6.77	1101322	0	576704	398266242	1209254052	4509452	0.12020597	0.1347143	0	0.13576361	0.13865676	0.54521226	0.13524341
80000	14.79	2363008	0	1233709	1604925631	1855272297	9660044	0.26260653	0.28904441	0	0.29043112	0.55875635	0.8364803	0.2897153
160000	33.72	5046625	0	2627510	2097423099	2110730899	20600986	0.59872159	0.61730589	0	0.61854998	0.7302198	0.95165806	0.61784614
256000	57.74	8410576	0	4372009	2897325801	2293606843	34310038	1.02521307	1.02878618	0	1.0292277	1.00870666	1.03411071	1.02899563

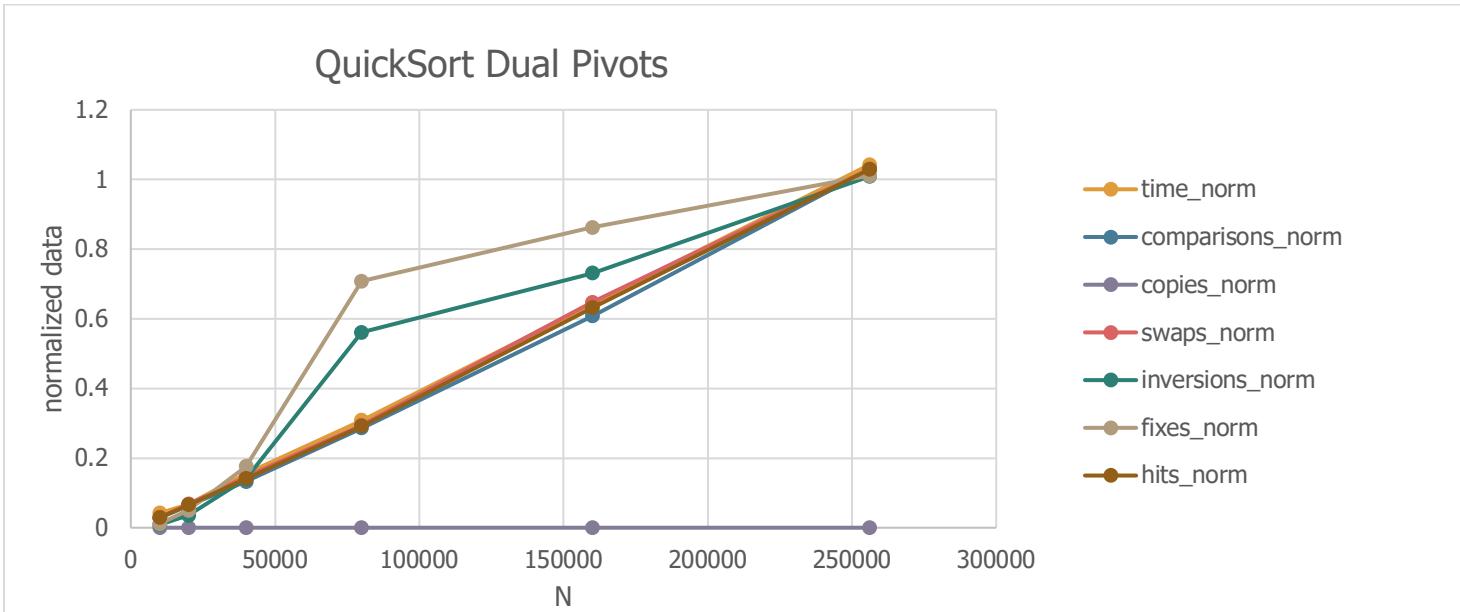
QuickSort Dual Pivots														
N	time	comparisons	copies	swaps	inversions	fixes	hits	time_norm	comparisons_norm	copies_norm	swaps_norm	inversions_norm	fixes_norm	hits_norm
10000	1.35	151787		62014	24835299	26745524	402765	0.04197761	0.02824882	0	0.02913346	0.00870407	0.01142627	0.02886198
20000	2.14	338489		143257	99832908	116888905	917149	0.06654229	0.06299562	0	0.06730047	0.03498861	0.04993747	0.06572252
40000	5.01	712249		312090	399394145	412346524	1971360	0.15578358	0.13255546	0	0.14661626	0.13997635	0.17616335	0.14216685
80000	9.92	1534839		626596	1600340564	1657079617	4063460	0.30845771	0.2856463	0	0.29436752	0.5608741	0.70794026	0.29118588
160000	20.67	3265825		1377444	2086294638	2019215870	8819483	0.64272388	0.60779719	0	0.64710718	0.73118726	0.8626527	0.63200055
256000	33.51	5525002		2190632	2878132621	2367450977	14357631	1.04197761	1.02824882	0	1.02913346	1.00870407	1.01142627	1.02886198

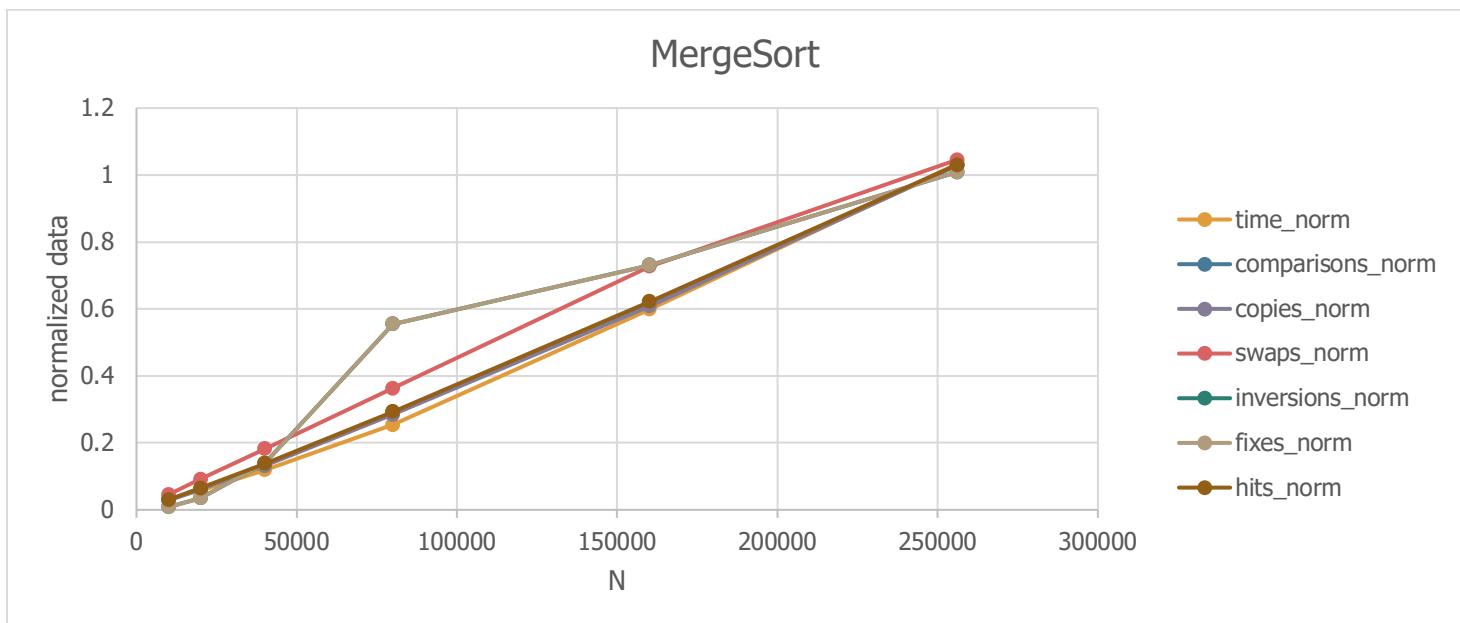
MergeSort														
N	time	comparisons	copies	swaps	inversions	fixes	hits	time_norm	comparisons_norm	copies_norm	swaps_norm	inversions_norm	fixes_norm	hits_norm
10000	1.51	123551	109952	9756	24977728	24977728	273506	0.03275488	0.02915491	0.02790658	0.04547807	0.00867139	0.00867139	0.03005111
20000	2.65	267137	239904	19700	100301422	100301422	588108	0.05748373	0.06303758	0.06088931	0.0918325	0.03482114	0.03482114	0.0646176
40000	5.48	574123	519843	39076	400388388	400388388	1254976	0.11887202	0.13547852	0.13193978	0.18215466	0.13900081	0.13900081	0.13788884
80000	11.69	1228303	1119683	77917	1599495509	1599495509	2671284	0.25357918	0.28984846	0.28418336	0.36321386	0.55528875	0.55528875	0.29350383
160000	27.6	2616355	2399350	155942	2105372485	2105372485	5663610	0.59869848	0.61739365	0.60897177	0.72693116	0.7309115	0.7309115	0.62228172
256000	47.61	4361293	4049954	224277	2905452988	2905452988	9374866	1.03275488	1.02915491	1.02790658	1.04547807	1.00867139	1.00867139	1.03005111



Because each instrumentation is different, we need to normalize each value before drawing. We can see from the figure that the curve of comparisons, swaps, hits and time is the closest, so the best predictor of total execution time is comparisons, swaps and hits.



Because each instrumentation is different, we need to normalize each value before drawing. We can see from the figure that the curve of comparisons, swaps, hits and time is the closest, so the best predictor of total execution time is swaps and hits.



Because each instrumentation is different, we need to normalize each value before drawing. We can see from the figure that the curve of comparisons, swaps, hits, copies and time is the closest, so the best predictor of total execution time is comparisons, swaps, copies and hits.