

Program Structures and Algorithms
Spring 2024

NAME: Gautham Venkata Krishna Prasad

NUID: 002249901

GITHUB LINK: <https://github.com/gauthamkris7neu/INFO6205Assignment>

Task:

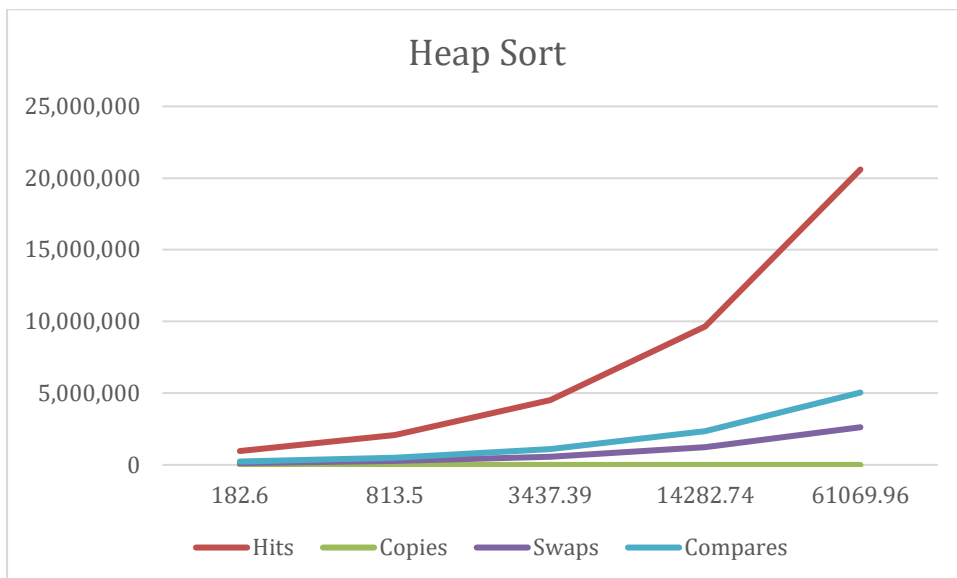
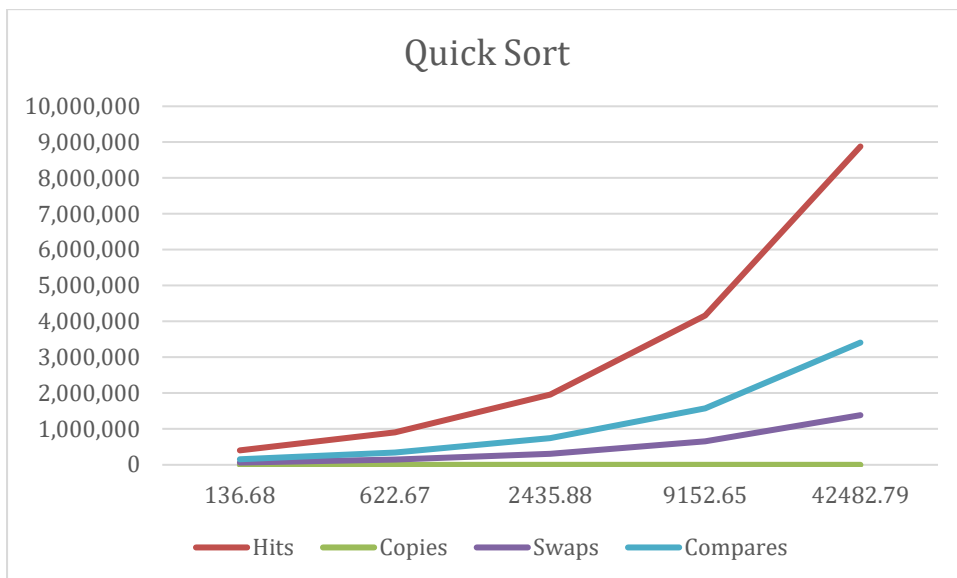
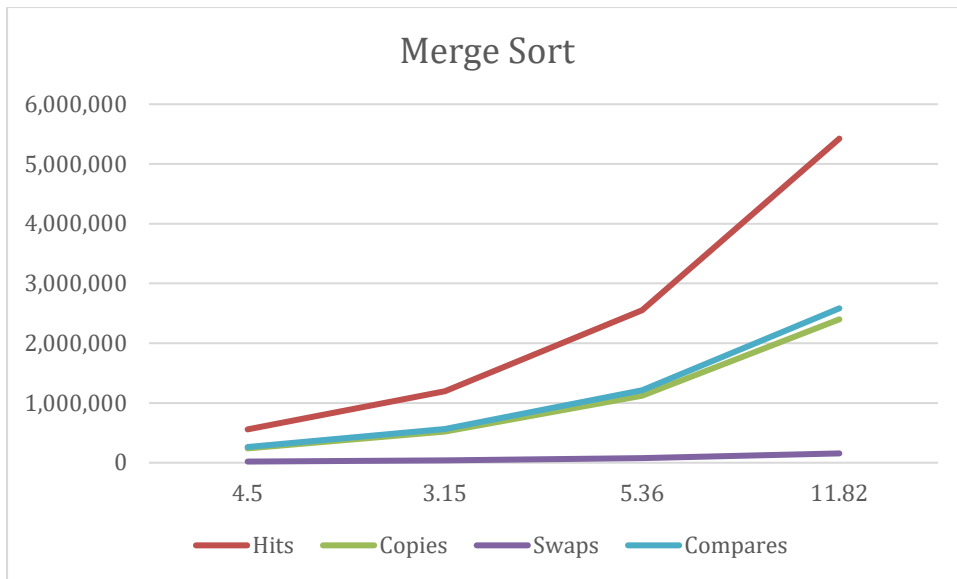
Assignment 6 : Determine for sorting algorithms , What is the best predictor of total execution time: comparisons, swaps/copies, hits (array accesses), memory used, or some combination of these.

Relationship Conclusion:

When we do a regression analysis with the below data it reveals a high coefficient of determination (R^2) value of approximately 0.961, suggesting a strong relationship between the chosen metrics and the instrumented run time of sorting algorithms. Specifically, 'Hits' (array accesses) have the most significant positive correlation with run time, indicating they are a strong predictor of performance. 'Copies' and 'Swaps' have a negative correlation with run time. 'Compares' also show a positive correlation, but to lesser extent than 'Hits.' Overall, while 'Hits' seem to be the best single predictor, the combination of all the metrics provides a robust prediction of the sorting algorithms execution time.

Evidence to support that conclusion:

Sorting Algorithm	Array Size	Instrumented Run Time(mSec)	Hits	Copies	Swaps	Compares	Run Time(mSec)
Merge Sort	10000	4.5	259,064	110,000	9,766	121,508	4
	20000	3.15	558,078	240,000	19,520	262,982	3
	40000	5.36	1,196,508	520,000	39,127	566,038	5.45
	80000	11.82	2,552,116	1,120,000	78,029	1,211,946	13.19
	160000	28.53	5,424,472	2,400,000	156,118	2,583,975	28.4
Quick Sort Dual Pivot	10000	136.68	403,231	0	63,224	154,723	2
	20000	622.67	902,065	0	142,565	340,465	3.64
	40000	2435.88	1,958,361	0	307,241	746,867	3.87
	80000	9152.65	4,160,868	0	655,659	1,572,976	7.19
	160000	42482.79	8,879,390	0	1,384,792	3,409,662	18.62
Heap Sort	10000	182.6	967,353	0	124,178	235,320	1
	20000	813.5	2,095,179	0	268,410	510,769	2.52
	40000	3437.39	4,510,220	0	576,811	1,101,488	5.59
	80000	14282.74	9,660,877	0	1,233,683	2,363,073	12.83
	160000	61069.96	20,599,917	0	2,627,058	5,045,842	30.12



Unit Test Screenshots:

```

2024-03-15 01:46:18 WARN SortBenchmark - No word counts specified on the command line
2024-03-15 01:46:18 INFO SortBenchmark-main: null with minimum: 10000 and maximum: 254000 and strategy: null
2024-03-15 01:46:18 INFO SortBenchmark - Beginning String sorts
2024-03-15 01:46:18 INFO SortBenchmark - Beginning String sorts
2024-03-15 01:46:18 INFO SortBenchmark - Beginning LocalByteLine sorts
2024-03-15 01:46:18 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.Integer from 10,000 total elements and 10 runs using sorter: MergeSort
2024-03-15 01:46:18 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with 10,000 elements with 10 runs
2024-03-15 01:46:18 INFO Timelogger - Raw time per run (nSec): 4.50
2024-03-15 01:46:18 INFO Timelogger - Normalized time per run (n log n): 6.33
2024-03-15 01:46:18 INFO SorterBenchmark - MergeSort: StackPack {hits: mean=259,046; stdDev=395, normalized=1.813; copies: 110,000, normalized=1.194; inversions: <unset>; swaps: mean=9,766; stdDev=76, normalized=0.106; fixes: mean=24,977,764;
2024-03-15 01:46:18 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.Integer from 10,000 total elements and 10 runs using sorter: HeapSort
2024-03-15 01:46:18 INFO Benchmark_Timer - Begin run: Instrumenting helper for HeapSort with 10,000 elements with 10 runs
2024-03-15 01:46:18 INFO Timelogger - Raw time per run (nSec): 182.60
2024-03-15 01:46:18 INFO Timelogger - Normalized time per run (n log n): 256.92
2024-03-15 01:46:18 INFO SorterBenchmark - HeapSort: StackPack {hits: mean=937,353; stdDev=394, normalized=1.583; copies: 0, normalized=0.000; inversions: <unset>; swaps: mean=124,178; stdDev=58, normalized=1.348; fixes: mean=75,583,977;
2024-03-15 01:46:18 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.Integer from 10,000 total elements and 10 runs using sorter: QuickSort_DualPivot
2024-03-15 01:46:18 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort_DualPivot with 10,000 elements with 10 runs
2024-03-15 01:46:18 INFO Timelogger - Raw time per run (nSec): 176.45

```

```

Run SortBenchmark
C:\Program Files\Java\jdk-17\bin\java.exe
2024-03-15 03:28:12 WARN SortBenchmark - No word counts specified on the command line
2024-03-15 03:28:12 INFO SortBenchmark - SortBenchmark.main: null with minimum: 10000 and maximum: 256000 and strategy: null
2024-03-15 03:28:12 INFO SortBenchmark - Beginning String sorts
2024-03-15 03:28:12 INFO SortBenchmark - Beginning String sorts
2024-03-15 03:28:12 INFO SortBenchmark - Beginning LocalDate sorts
2024-03-15 03:28:12 INFO SortBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.Integer from 10,000 total elements and 10 runs using sorter: MergeSort
2024-03-15 03:28:12 INFO Benchmark.Timer - Begin run: Helper for MergeSort with 10000 elements with 10 runs
2024-03-15 03:28:12 INFO TimerLogger - Raw time per run (secs): 3.76
2024-03-15 03:28:12 INFO TimerLogger - Normalized time per run (n log n): 5.32
2024-03-15 03:28:12 INFO SortBenchmark -
2024-03-15 03:28:12 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.Integer from 10,000 total elements and 10 runs using sorter: HeapSort
2024-03-15 03:28:12 INFO Benchmark.Timer - Begin run: Helper for HeapSort with 10000 elements with 10 runs
2024-03-15 03:28:12 INFO TimerLogger - Raw time per run (secs): 1.45
2024-03-15 03:28:12 INFO TimerLogger - Normalized time per run (n log n): 2.04
2024-03-15 03:28:12 INFO SortBenchmark -
2024-03-15 03:28:12 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.Integer from 10,000 total elements and 10 runs using sorter: QuickSortDualPivot
2024-03-15 03:28:12 INFO Benchmark.Timer - Begin run: Helper for QuickSortDualPivot with 10000 elements with 10 runs
2024-03-15 03:28:12 INFO TimerLogger - Raw time per run (secs): 1.55
2024-03-15 03:28:12 INFO TimerLogger - Normalized time per run (n log n): 2.18
2024-03-15 03:28:12 INFO SortBenchmark -
2024-03-15 03:28:12 INFO SorterBenchmark - run: sort 20,000 elements using SorterBenchmark on class java.lang.Integer from 20,000 total elements and 10 runs using sorter: MergeSort
info205Assignment > src > main > java > edu > nau > cse > info205 > util > @ SortBenchmark
2320 CRLF UTF-8 4 space

```

```

Run MergeSortTest
✓ MergeSortTest (https://en.cppreference.com/algorithm/linearithmic) 214 ms
  ✓ testSort11_partialsorted 86 ms
  ✓ testSort9_partialsorted 30 ms
  ✓ testSort1 2 ms
  ✓ testSort2 8 ms
  ✓ testSort3 2 ms
  ✓ testSort4 27 ms
  ✓ testSort5 8 ms
  ✓ testSort6 8 ms
  ✓ testSort7 7 ms
  ✓ testSort10_partialsorted 14 ms
  ✓ testSort8_partialsorted 17 ms
  ✓ testSort12 4 ms
  ✓ testSort13 1 ms
  ✓ testSort14 0 ms
  ✓ testSort1a 0 ms

✓ Tests passed: 15 of 15 tests - 214 ms
  EchoProgram Files\Visual\jdk-17\bin\java.exe
  Instrumenting helper for insertion sort with 128 elements
  partial sorted average time partialSorted.Cutoff + Insurance + NoCopy: 62888
  Instrumenting helper for insertion sort with 128 elements
  partial sorted average time partialSorted.Cutoff + NoCopy: 29028
  Instrumenting helper for merge sort with 128 elements
  StatPack (hits: 1,684, normalized=2.711; copies: 640, normalized=1.030; inversions: 4,224, normalized=6.801; swaps: 101, normalized=0.163; fixes: 4,224, normalized=6.801; compares=751)
  Worst Compares=769
  Instrumenting helper for insertion sort with 128 elements
  Instrumenting helper for merge sort with 128 elements
  StatPack (hits: 1,792, normalized=2.885; copies: 896, normalized=1.443; inversions: <unset>; swaps: 0, normalized=0.000; fixes: 0, normalized=0.000; compares: 448, no
  Instrumenting helper for insertion sort with 128 elements
  average time random.Cutoff: 25327
  Instrumenting helper for insertion sort with 128 elements
  average time random.Cutoff + NoCopy: 7017

```