Program Structures and Algorithms
Fall 2024

NAME: Sanskruti Manoria
NUID: 002643300
GITHUB LINK: https://github.com/sannskruti/INFO6205

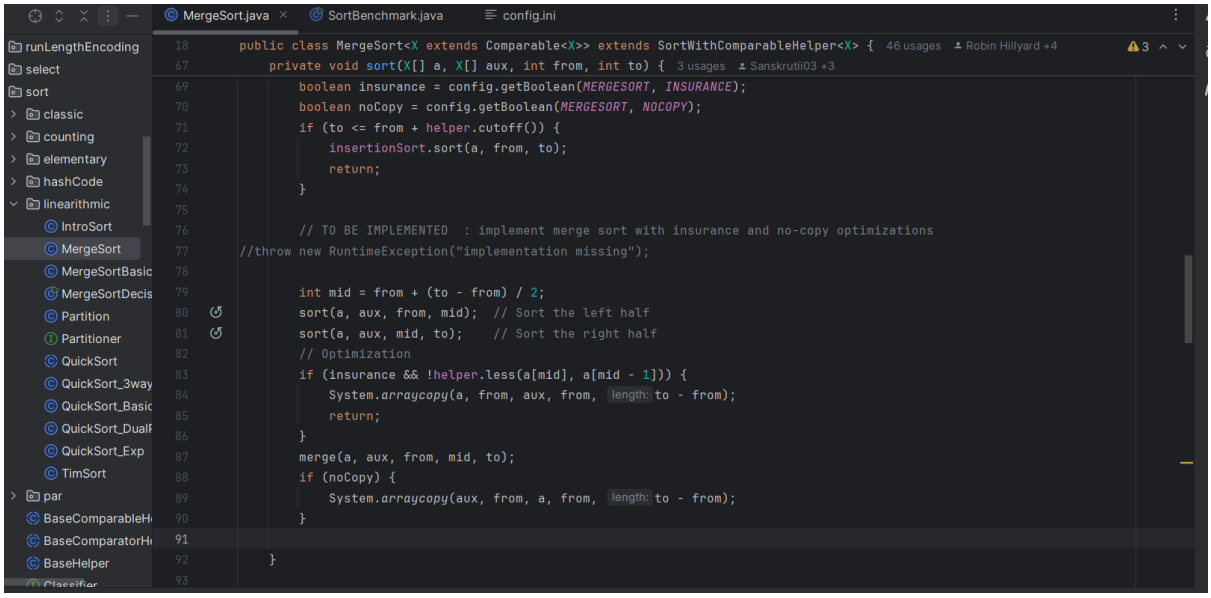
Assignment 5 Hits Predictor

In this assignment, your task is to 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.

You will run the benchmarks for merge sort, (dual-pivot) quick sort, and heap sort. You will sort randomly generated arrays of between 10,000 and 256,000 elements (doubling the size each time). If you use the *SortBenchmark*, as I expect, the number of runs is chosen for you. So, you can ignore the instructions about setting the number of runs.

Code:

Mergesort.java



SortBenchmark: Added stats for heap sort

```
public class SortBenchmark {

    private void runIntegerSorts(long N) {
        if (N > Integer.MAX_VALUE) throw new SortException("number of elements is too large");
        int n = (int) N;
        if (isConfigBenchmarkIntegerSorter("shellsort"))
            sortIntegersByShellSort(n);
        if (isConfigBenchmarkIntegerSorter("bucketsort"))
            runIntegerBucketSort(n);
        if (isConfigBenchmarkIntegerSorter("quicksort"))
            runIntegerQuickSort(n);
        if (isConfigBenchmarkIntegerSorter("heapsort"))
            runIntegerHeapSort(n);
```

```
if (isConfigBenchmarkStringSorter("heapsort") && nRunsLinearithmic > 0) {
    Helper<String> helper = HelperFactory.create("Heapsort", nWords, config);
    try (SortWithHelper<String> sorter = new HeapSort<>(nWords, config)) {
        runStringSortBenchmark(words, nWords, nRuns: nRunsLinearithmic * 3, sorter, timeLoggersLinearithmic);
    }
}
```

```
private void runIntegerHeapSort(int N) {
    SortWithHelper<Integer> sorter = new HeapSort<>(N, config);
    Integer[] numbers = sorter.getHelper().random(Integer.class, Random::nextInt);
    int runs = config.getInt(BENCHMARKINTEGERSORTERS, optionName: "runs", defaultValue: 1000);
    runIntegerSortBenchmark(numbers, N, runs, sorter, sorter::preProcess, timeLoggersLinearithmic);
}
```

Config.ini

MergeSort.java SortBenchmark.java config.ini

Plugins supporting *.ini files found.

```ini
[sortbenchmark]
version = 1.0.0 (sortbenchmark)

[helper]
instrument = true
seed =
cutoff =

[instrumenting]
# The options in this section apply only if instrument (in [helper]) is set to true.
# this normally should be true
showStats = true
swaps = true
compares = true
copies = true
hits = true
# The following settings slow everything down a lot so keep fixes false and inversions small (or z
fixes = false
inversions = 0

[benchmarkstringsorters]
words = 1000
runs = 1000
mergesort = true
timsort = false
```

Plugins supporting *.ini files found.

```ini
heapsort=true

[benchmarkdatesorters]
timsort = false
n = 100000

[mergesort]
insurance = false
nocopy = true

[benchmarkintegersorters]
shellsort = true
mode = 3
runs =

[operationsbenchmark]
nlargest = 10000000
repetitions = 10
```

Output:

Taken array size 10000,20000,40000,80000,160000,256000 for all 3 sort (instrument =true)

```
C:\Users\Dell\.jdks\openjdk-23\bin\java.exe "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.3\lib\idea_rt.jar=60962:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2024.2.3\bin" -Dfile.encoding=UTF-8 -Dsun.stdout.encoding=UTF-8 -Dsun.stderr.encoding=UTF-8 -classpath D:\Sanskruti\INFO6205\target\classes;C:\Users\Dell\.m2\repository\com\phasmidsoftware\args_2.13\1.0.3\args_2.13-1.0.3.jar;C:\Users\Dell\.m2\repository\org\scala-lang\scala-library\2.13.7\scala-library-2.13.7.jar;C:\Users\Dell\.m2\repository\org\scala-lang\modules\scala-parser-combinators_2.13\1.1.2\scala-parser-combinators_2.13-1.1.2.jar;C:\Users\Dell\.m2\repository\com\phasmidsoftware\number_2.13\1.0.12\number_2.13-1.0.12.jar;C:\Users\Dell\.m2\repository\com\phasmidsoftware\flog_2.13\1.0.8\flog_2.13-1.0.8.jar;C:\Users\Dell\.m2\repository\org\slf4j\slf4j-api\1.7.30\slf4j-api-1.7.30.jar;C:\Users\Dell\.m2\repository\com\phasmidsoftware\matchers_2.13\1.0.5\matchers_2.13-1.0.5.jar;C:\Users\Dell\.m2\repository\org\scala-lang\scala-compiler\2.13.6\scala-compiler-2.13.6.jar;C:\Users\Dell\.m2\repository\org\scala-lang\scala-reflect\2.13.6\scala-reflect-2.13.6.jar;C:\Users\Dell\.m2\repository\org\jline\jline\3.19.0\jline-3.19.0.jar;C:\Users\Dell\.m2\repository\net\java\dev\jna\jna\5.3.1\jna-5.3.1.jar;C:\Users\Dell\.m2\repository\org\apache\commons\commons-math3\3.6.1\commons-math3-3.6.1.jar;C:\Users\Dell\.m2\repository\org\apache\logging\log4j\log4j-api\2.19.0\log4j-api-2.19.0.jar;C:\Users\Dell\.m2\repository\log4j\log4j\1.2.17\log4j-1.2.17.jar;C:\Users\Dell\.m2\repository\com\google\guava\guava\31.1-jre\guava-31.1-jre.jar;C:\Users\Dell\.m2\repository\com\google\guava\failureaccess\1.0.1\failureaccess-1.0.1.jar;C:\Users\Dell\.m2\repository\com\google\guava\listenablefuture\9999.0-empty-to-avoid-conflict-with-guava\listenablefuture-9999.0-empty-to-avoid-conflict-with-guava.jar;C:\Users\Dell\.m2\repository\com\google\code\findbugs\jsr305\3.0.2\jsr305-3.0.2.jar;C:\Users\Dell\.m2\repository\org\checkerframework\checker-qual\3.12.0\checker-qual-3.12.0.jar;C:\Users\Dell\.m2\repository\com\google\errorprone\error_prone_annotations\2.11.0\error_prone_annotations-2.11.0.jar;C:\Users\Dell\.m2\repository\com\google\j2objc\j2objc-annotations\1.3\j2objc-annotations-1.3.jar;C:\Users\Dell\.m2\repository\org\ini4j\ini4j\0.5.4\ini4j-0.5.4.jar edu.neu.coe.info6205.util.SortBenchmark 10000 20000 40000 80000 160000 256000
2024-11-22 12:59:55.815 INFO  SortBenchmark - !!!!!!!!!!!!!!!!!!!! SortBenchmark Start !!!!!!!!!!!!!!!!!!!!

2024-11-22 12:59:55.821 INFO  SortBenchmark - SortBenchmark.main: version 1.0.0 (sortbenchmark) with word counts: [10000, 20000, 40000, 80000, 160000, 256000]
2024-11-22 12:59:55.821 INFO  SortBenchmark - Beginning String sorts
2024-11-22 12:59:55.824 INFO  SortBenchmark - ############################## 10000 words ##############################
2024-11-22 12:59:56.074 INFO  SortBenchmarkHelper - Testing with words: 22,865 from eng-uk_web_2002_10K-sentences.txt
2024-11-22 12:59:56.082 INFO  SortBenchmark - benchmarkStringSorters: sorting 10,000 words and instrumented with total work (for estimating runs): 1.0E8
2024-11-22 12:59:56.097 INFO  SortBenchmark - ****************************** String sort: 3376 runs of 10000 MergeSort with no copy ******************************
2024-11-22 12:59:56.105 INFO  SorterBenchmark - run: sort 10,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 3,376 runs
2024-11-22 12:59:56.106 INFO  Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with no copy with 10,000 elements with 3,376 runs
2024-11-22 13:00:06.260 INFO  TimeLogger - 10000@MergeSort with no copy: Raw time per run {mSec}:  2.5441
2024-11-22 13:00:06.261 INFO  TimeLogger - 10000@MergeSort with no copy: Normalized time per run {n log n}: 21.4771
2024-11-22 13:00:06.265 INFO  InstrumentedComparableHelper - 10000@MergeSort with no copy: StatPack {runs: 3376 hits: mean=235035; stdDev=325; normalized=2.552; lookups: <unset>; copies: mean=100000; normalized=1.086; inversions: <unset>; swaps: mean=46352; stdDev=336; normalized=0.503; fixes: <unset>; compares: mean=143533; stdDev=326; normalized=1.558}
```

2024-11-22 13:00:06.265 INFO SortBenchmark -
********** (10.174 sec.)
2024-11-22 13:00:06.266 INFO SortBenchmark - ********** String sort: 3376 runs of 10000 QuickSort dual pivot **********
2024-11-22 13:00:06.267 INFO SorterBenchmark - run: sort 10,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 3,376 runs
2024-11-22 13:00:06.267 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 10,000 elements with 3,376 runs
2024-11-22 13:00:14.726 INFO TimeLogger - 10000@QuickSort dual pivot: Raw time per run {mSec}: 2.1211
2024-11-22 13:00:14.727 INFO TimeLogger - 10000@QuickSort dual pivot: Normalized time per run {n log n}: 17.9063
2024-11-22 13:00:14.727 INFO InstrumentedComparableHelper - 10000@QuickSort dual pivot: StatPack {runs: 1 hits: mean=141343; stdDev=5401; normalized=1.535; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=76356; stdDev=3967; normalized=0.829; fixes: <unset>; compares: mean=163120; stdDev=6351; normalized=1.771}
2024-11-22 13:00:14.728 INFO SortBenchmark -
********** (8.462 sec.)
2024-11-22 13:00:14.728 INFO SortBenchmark - ********** String sort: 2532 runs of 10000 Heap Sort **********
2024-11-22 13:00:14.728 INFO SorterBenchmark - run: sort 10,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 2,532 runs
2024-11-22 13:00:14.728 INFO Benchmark_Timer - Begin run: Instrumenting helper for Heap Sort with 10,000 elements with 2,532 runs
2024-11-22 13:00:24.023 INFO TimeLogger - 10000@Heap Sort: Raw time per run {mSec}: 3.2532
2024-11-22 13:00:24.023 INFO TimeLogger - 10000@Heap Sort: Normalized time per run {n log n}: 27.4625
2024-11-22 13:00:24.024 INFO InstrumentedComparableHelper - 10000@Heap Sort: StatPack {runs: 1 hits: mean=719151; stdDev=325; normalized=7.808; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=124204; stdDev=76; normalized=1.349; fixes: <unset>; compares: mean=235371; stdDev=95; normalized=2.556}
2024-11-22 13:00:24.024 INFO SortBenchmark -
********** (9.296 sec.)
2024-11-22 13:00:24.025 INFO SortBenchmark - ########## 20000 words ##########
2024-11-22 13:00:24.163 INFO SortBenchmarkHelper - Testing with words: 22,865 from eng-uk_web_2002_10K-sentences.txt
2024-11-22 13:00:24.163 INFO SortBenchmark - benchmarkStringSorters: sorting 20,000 words and instrumented with total work (for estimating runs): 1.0E8
2024-11-22 13:00:24.164 INFO SortBenchmark - ********** String sort: 1556 runs of 20000 MergeSort with no copy **********
2024-11-22 13:00:24.164 INFO SorterBenchmark - run: sort 20,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 1,556 runs
2024-11-22 13:00:24.164 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with no copy with 20,000 elements with 1,556 runs
2024-11-22 13:00:33.481 INFO TimeLogger - 20000@MergeSort with no copy: Raw time per run {mSec}: 5.3156
2024-11-22 13:00:33.481 INFO TimeLogger - 20000@MergeSort with no copy: Normalized time per run {n log n}: 20.6904
2024-11-22 13:00:33.482 INFO InstrumentedComparableHelper - 20000@MergeSort with no copy: StatPack {runs: 1556 hits: mean=510081; stdDev=465; normalized=2.575; lookups: <unset>; copies: mean=220000; normalized=1.111; inversions: <unset>; swaps: mean=92718; stdDev=479; normalized=0.468; fixes: <unset>; compares: mean=307077; stdDev=467; normalized=1.550}
2024-11-22 13:00:33.482 INFO SortBenchmark -
********** (9.318 sec.)
2024-11-22 13:00:33.482 INFO SortBenchmark - ********** String sort: 1556 runs of 20000 QuickSort dual pivot **********
2024-11-22 13:00:33.482 INFO SorterBenchmark - run: sort 20,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 1,556 runs
2024-11-22 13:00:33.482 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 20,000 elements with 1,556 runs
2024-11-22 13:00:41.570 INFO TimeLogger - 20000@QuickSort dual pivot: Raw time per run {mSec}: 4.5932

2024-11-22 13:00:41.570 INFO TimeLogger - 20000@QuickSort dual pivot: Normalized time per run {n log n}: 17.8787
2024-11-22 13:00:41.570 INFO InstrumentedComparableHelper - 20000@QuickSort dual pivot: StatPack {runs: 1 hits: mean=303654; stdDev=11030; normalized=1.533; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=161957; stdDev=7810; normalized=0.818; fixes: <unset>; compares: mean=353333; stdDev=13167; normalized=1.784}
2024-11-22 13:00:41.570 INFO SortBenchmark -
********** (8.088 sec.)
2024-11-22 13:00:41.571 INFO SortBenchmark - ********** String sort: 1167 runs of 20000 Heap Sort **********
2024-11-22 13:00:41.571 INFO SorterBenchmark - run: sort 20,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 1,167 runs
2024-11-22 13:00:41.571 INFO Benchmark_Timer - Begin run: Instrumenting helper for Heap Sort with 20,000 elements with 1,167 runs
2024-11-22 13:00:50.587 INFO TimeLogger - 20000@Heap Sort: Raw time per run {mSec}: 7.1054
2024-11-22 13:00:50.587 INFO TimeLogger - 20000@Heap Sort: Normalized time per run {n log n}: 27.6573
2024-11-22 13:00:50.588 INFO InstrumentedComparableHelper - 20000@Heap Sort: StatPack {runs: 1 hits: mean=1558307; stdDev=439; normalized=7.867; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=268406; stdDev=105; normalized=1.355; fixes: <unset>; compares: mean=510748; stdDev=129; normalized=2.579}
2024-11-22 13:00:50.588 INFO SortBenchmark -
********** (9.017 sec.)
2024-11-22 13:00:50.588 INFO SortBenchmark - ########## 40000 words ##########
2024-11-22 13:00:50.625 INFO SortBenchmarkHelper - Testing with words: 22,865 from eng-uk_web_2002_10K-sentences.txt
2024-11-22 13:00:50.625 INFO SortBenchmark - benchmarkStringSorters: sorting 40,000 words and instrumented with total work (for estimating runs): 1.0E8
2024-11-22 13:00:50.625 INFO SortBenchmark - ********** String sort: 724 runs of 40000 MergeSort with no copy **********
2024-11-22 13:00:50.626 INFO SorterBenchmark - run: sort 40,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 724 runs
2024-11-22 13:00:50.626 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with no copy with 40,000 elements with 724 runs
2024-11-22 13:01:00.149 INFO TimeLogger - 40000@MergeSort with no copy: Raw time per run {mSec}: 11.8066
2024-11-22 13:01:00.150 INFO TimeLogger - 40000@MergeSort with no copy: Normalized time per run {n log n}: 21.3189
2024-11-22 13:01:00.150 INFO InstrumentedComparableHelper - 40000@MergeSort with no copy: StatPack {runs: 724 hits: mean=1100165; stdDev=660; normalized=2.596; lookups: <unset>; copies: mean=480000; normalized=1.132; inversions: <unset>; swaps: mean=185434; stdDev=685; normalized=0.437; fixes: <unset>; compares: mean=654154; stdDev=665; normalized=1.543}
2024-11-22 13:01:00.150 INFO SortBenchmark -
********** (9.524 sec.)
2024-11-22 13:01:00.150 INFO SortBenchmark - ********** String sort: 724 runs of 40000 QuickSort dual pivot **********
2024-11-22 13:01:00.150 INFO SorterBenchmark - run: sort 40,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 724 runs
2024-11-22 13:01:00.151 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 40,000 elements with 724 runs
2024-11-22 13:01:08.251 INFO TimeLogger - 40000@QuickSort dual pivot: Raw time per run {mSec}: 9.9738
2024-11-22 13:01:08.252 INFO TimeLogger - 40000@QuickSort dual pivot: Normalized time per run {n log n}: 18.0094
2024-11-22 13:01:08.252 INFO InstrumentedComparableHelper - 40000@QuickSort dual pivot: StatPack {runs: 1 hits: mean=644950; stdDev=22936; normalized=1.522; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=337669; stdDev=16230; normalized=0.797; fixes: <unset>; compares: mean=755647; stdDev=24612; normalized=1.783}
2024-11-22 13:01:08.252 INFO SortBenchmark -
********** (8.103 sec.)
2024-11-22 13:01:08.252 INFO SortBenchmark - ********** String sort: 543 runs of 40000 Heap Sort **********

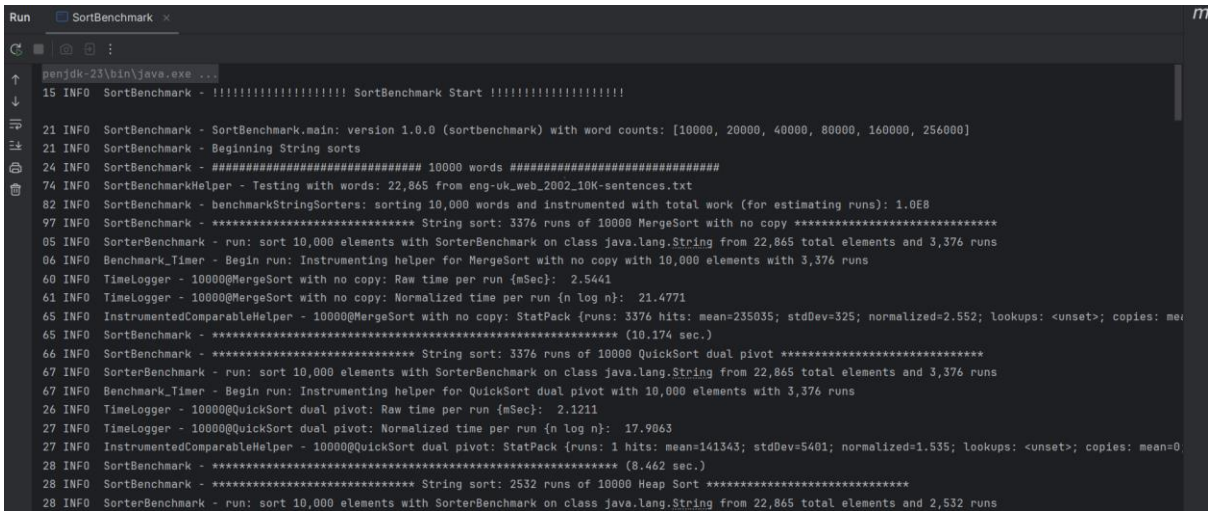
2024-11-22 13:01:08.253 INFO SorterBenchmark - run: sort 40,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 543 runs
2024-11-22 13:01:08.253 INFO Benchmark_Timer - Begin run: Instrumenting helper for Heap Sort with 40,000 elements with 543 runs
2024-11-22 13:01:22.301 INFO TimeLogger - 40000@Heap Sort: Raw time per run {mSec}: 23.9632
2024-11-22 13:01:22.301 INFO TimeLogger - 40000@Heap Sort: Normalized time per run {n log n}: 43.2697
2024-11-22 13:01:22.302 INFO InstrumentedComparableHelper - 40000@Heap Sort: StatPack {runs: 1 hits: mean=3356556; stdDev=635; normalized=7.919; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=576791; stdDev=151; normalized=1.361; fixes: <unset>; compares: mean=1101487; stdDev=185; normalized=2.599}
2024-11-22 13:01:22.302 INFO SortBenchmark - ********** (14.049 sec.)
2024-11-22 13:01:22.302 INFO SortBenchmark - ########## 80000 words ##########
2024-11-22 13:01:22.942 INFO SortBenchmarkHelper - Testing with words: 81,546 from eng-uk_web_2002_100K-sentences.txt
2024-11-22 13:01:22.944 INFO SortBenchmark - benchmarkStringSorters: sorting 80,000 words and instrumented with total work (for estimating runs): 1.0E8
2024-11-22 13:01:22.944 INFO SortBenchmark - ********** String sort: 336 runs of 80000 MergeSort with no copy **********
2024-11-22 13:01:22.944 INFO SorterBenchmark - run: sort 80,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 336 runs
2024-11-22 13:01:22.945 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with no copy with 80,000 elements with 336 runs
2024-11-22 13:01:37.351 INFO TimeLogger - 80000@MergeSort with no copy: Raw time per run {mSec}: 35.7202
2024-11-22 13:01:37.351 INFO TimeLogger - 80000@MergeSort with no copy: Normalized time per run {n log n}: 30.0774
2024-11-22 13:01:37.352 INFO InstrumentedComparableHelper - 80000@MergeSort with no copy: StatPack {runs: 336 hits: mean=2360482; stdDev=976; normalized=2.614; lookups: <unset>; copies: mean=1040000; normalized=1.151; inversions: <unset>; swaps: mean=371037; stdDev=1016; normalized=0.411; fixes: <unset>; compares: mean=1388458; stdDev=979; normalized=1.537}
2024-11-22 13:01:37.352 INFO SortBenchmark - ********** (14.407 sec.)
2024-11-22 13:01:37.352 INFO SortBenchmark - ********** String sort: 336 runs of 80000 QuickSort dual pivot **********
2024-11-22 13:01:37.352 INFO SorterBenchmark - run: sort 80,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 336 runs
2024-11-22 13:01:37.352 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 80,000 elements with 336 runs
2024-11-22 13:01:47.611 INFO TimeLogger - 80000@QuickSort dual pivot: Raw time per run {mSec}: 25.1696
2024-11-22 13:01:47.612 INFO TimeLogger - 80000@QuickSort dual pivot: Normalized time per run {n log n}: 21.1935
2024-11-22 13:01:47.612 INFO InstrumentedComparableHelper - 80000@QuickSort dual pivot: StatPack {runs: 1 hits: mean=1392760; stdDev=45470; normalized=1.542; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=735720; stdDev=33541; normalized=0.815; fixes: <unset>; compares: mean=1630481; stdDev=46900; normalized=1.805}
2024-11-22 13:01:47.612 INFO SortBenchmark - ********** (10.259 sec.)
2024-11-22 13:01:47.612 INFO SortBenchmark - ********** String sort: 252 runs of 80000 Heap Sort **********
2024-11-22 13:01:47.612 INFO SorterBenchmark - run: sort 80,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 252 runs
2024-11-22 13:01:47.612 INFO Benchmark_Timer - Begin run: Instrumenting helper for Heap Sort with 80,000 elements with 252 runs
2024-11-22 13:01:58.863 INFO TimeLogger - 80000@Heap Sort: Raw time per run {mSec}: 38.8730
2024-11-22 13:01:58.863 INFO TimeLogger - 80000@Heap Sort: Normalized time per run {n log n}: 32.7321
2024-11-22 13:01:58.863 INFO InstrumentedComparableHelper - 80000@Heap Sort: StatPack {runs: 1 hits: mean=7193158; stdDev=928; normalized=7.964; lookups: <unset>; copies: mean=0; normalized=0.000; inversions:

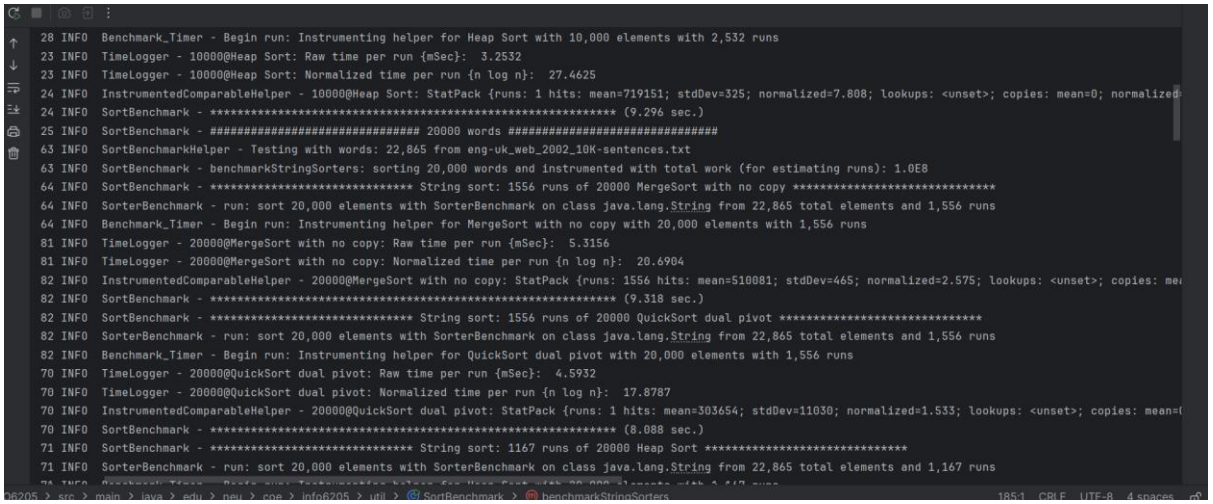
<unset>; swaps: mean=1233610; stdDev=215; normalized=1.366; fixes: <unset>; compares: mean=2362969; stdDev=273; normalized=2.616}
2024-11-22 13:01:58.863 INFO SortBenchmark -
********** (11.252 sec.)
2024-11-22 13:01:58.863 INFO SortBenchmark - ########## 160000 words ##########
2024-11-22 13:01:59.281 INFO SortBenchmarkHelper - Testing with words: 81,546 from eng-uk_web_2002_100K-sentences.txt
2024-11-22 13:01:59.282 INFO SortBenchmark - benchmarkStringSorters: sorting 160,000 words and instrumented with total work (for estimating runs): 1.0E8
2024-11-22 13:01:59.282 INFO SortBenchmark - ********** String sort: 156 runs of 160000 MergeSort with no copy **********
2024-11-22 13:01:59.283 INFO SorterBenchmark - run: sort 160,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 156 runs
2024-11-22 13:01:59.283 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with no copy with 160,000 elements with 156 runs
2024-11-22 13:02:13.752 INFO TimeLogger - 160000@MergeSort with no copy: Raw time per run {mSec}: 80.2564
2024-11-22 13:02:13.752 INFO TimeLogger - 160000@MergeSort with no copy: Normalized time per run {n log n}: 31.6567
2024-11-22 13:02:13.752 INFO InstrumentedComparableHelper - 160000@MergeSort with no copy: StatPack {runs: 156 hits: mean=5040520; stdDev=1320; normalized=2.629; lookups: <unset>; copies: mean=2240000; normalized=1.168; inversions: <unset>; swaps: mean=741622; stdDev=1364; normalized=0.387; fixes: <unset>; compares: mean=2936457; stdDev=1339; normalized=1.532}
2024-11-22 13:02:13.753 INFO SortBenchmark -
********** (14.471 sec.)
2024-11-22 13:02:13.753 INFO SortBenchmark - ********** String sort: 156 runs of 160000 QuickSort dual pivot **********
2024-11-22 13:02:13.753 INFO SorterBenchmark - run: sort 160,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 156 runs
2024-11-22 13:02:13.753 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 160,000 elements with 156 runs
2024-11-22 13:02:23.740 INFO TimeLogger - 160000@QuickSort dual pivot: Raw time per run {mSec}: 53.9487
2024-11-22 13:02:23.740 INFO TimeLogger - 160000@QuickSort dual pivot: Normalized time per run {n log n}: 21.2798
2024-11-22 13:02:23.741 INFO InstrumentedComparableHelper - 160000@QuickSort dual pivot: StatPack {runs: 1 hits: mean=2932846; stdDev=91243; normalized=1.530; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=1525433; stdDev=69543; normalized=0.796; fixes: <unset>; compares: mean=3471582; stdDev=101011; normalized=1.811}
2024-11-22 13:02:23.741 INFO SortBenchmark -
********** (9.988 sec.)
2024-11-22 13:02:23.741 INFO SortBenchmark - ********** String sort: 117 runs of 160000 Heap Sort **********
2024-11-22 13:02:23.741 INFO SorterBenchmark - run: sort 160,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 117 runs
2024-11-22 13:02:23.741 INFO Benchmark_Timer - Begin run: Instrumenting helper for Heap Sort with 160,000 elements with 117 runs
2024-11-22 13:02:35.799 INFO TimeLogger - 160000@Heap Sort: Raw time per run {mSec}: 91.0684
2024-11-22 13:02:35.799 INFO TimeLogger - 160000@Heap Sort: Normalized time per run {n log n}: 35.9214
2024-11-22 13:02:35.800 INFO InstrumentedComparableHelper - 160000@Heap Sort: StatPack {runs: 1 hits: mean=15346374; stdDev=1241; normalized=8.004; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=2627210; stdDev=307; normalized=1.370; fixes: <unset>; compares: mean=5045977; stdDev=353; normalized=2.632}
2024-11-22 13:02:35.800 INFO SortBenchmark -
********** (12.059 sec.)
2024-11-22 13:02:35.800 INFO SortBenchmark - ########## 256000 words ##########
2024-11-22 13:02:36.243 INFO SortBenchmarkHelper - Testing with words: 81,546 from eng-uk_web_2002_100K-sentences.txt

2024-11-22 13:02:36.244 INFO SortBenchmark - benchmarkStringSorters: sorting 256,000 words and instrumented with total work (for estimating runs): 1.0E8
2024-11-22 13:02:36.245 INFO SortBenchmark - ********** String sort: 96 runs of 256000 MergeSort with no copy **********
2024-11-22 13:02:36.245 INFO SorterBenchmark - run: sort 256,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 96 runs
2024-11-22 13:02:36.245 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with no copy with 256,000 elements with 96 runs
2024-11-22 13:02:52.723 INFO TimeLogger - 256000@MergeSort with no copy: Raw time per run {mSec}: 148.1979
2024-11-22 13:02:52.723 INFO TimeLogger - 256000@MergeSort with no copy: Normalized time per run {n log n}: 35.0356
2024-11-22 13:02:52.724 INFO InstrumentedComparableHelper - 256000@MergeSort with no copy: StatPack {runs: 96 hits: mean=8322317; stdDev=1155; normalized=2.611; lookups: <unset>; copies: mean=3840000; normalized=1.205; inversions: <unset>; swaps: mean=936940; stdDev=1210; normalized=0.294; fixes: <unset>; compares: mean=4690474; stdDev=1149; normalized=1.471}
2024-11-22 13:02:52.724 INFO SortBenchmark - ********** (16.479 sec.)
2024-11-22 13:02:52.724 INFO SortBenchmark - ********** String sort: 96 runs of 256000 QuickSort dual pivot **********
2024-11-22 13:02:52.724 INFO SorterBenchmark - run: sort 256,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 96 runs
2024-11-22 13:02:52.725 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 256,000 elements with 96 runs
2024-11-22 13:03:04.310 INFO TimeLogger - 256000@QuickSort dual pivot: Raw time per run {mSec}: 101.9271
2024-11-22 13:03:04.310 INFO TimeLogger - 256000@QuickSort dual pivot: Normalized time per run {n log n}: 24.0967
2024-11-22 13:03:04.311 INFO InstrumentedComparableHelper - 256000@QuickSort dual pivot: StatPack {runs: 1 hits: mean=4845453; stdDev=138452; normalized=1.520; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=2477435; stdDev=98507; normalized=0.777; fixes: <unset>; compares: mean=5766939; stdDev=160660; normalized=1.809}
2024-11-22 13:03:04.311 INFO SortBenchmark - ********** (11.587 sec.)
2024-11-22 13:03:04.311 INFO SortBenchmark - ********** String sort: 72 runs of 256000 Heap Sort **********
2024-11-22 13:03:04.311 INFO SorterBenchmark - run: sort 256,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 72 runs
2024-11-22 13:03:04.311 INFO Benchmark_Timer - Begin run: Instrumenting helper for Heap Sort with 256,000 elements with 72 runs
2024-11-22 13:03:17.937 INFO TimeLogger - 256000@Heap Sort: Raw time per run {mSec}: 166.2361
2024-11-22 13:03:17.937 INFO TimeLogger - 256000@Heap Sort: Normalized time per run {n log n}: 39.3000
2024-11-22 13:03:17.938 INFO InstrumentedComparableHelper - 256000@Heap Sort: StatPack {runs: 1 hits: mean=25564796; stdDev=1592; normalized=8.019; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=4371968; stdDev=382; normalized=1.371; fixes: <unset>; compares: mean=8410429; stdDev=453; normalized=2.638}
2024-11-22 13:03:17.938 INFO SortBenchmark - ********** (13.627 sec.)
2024-11-22 13:03:17.941 INFO SortBenchmark - ********** Integer sort: 10000 Shell sort in mode 3 **********
2024-11-22 13:03:17.941 INFO SorterBenchmark - run: sort 10,000 elements with SorterBenchmark on class java.lang.Integer from 10,000 total elements and 1,000 runs
2024-11-22 13:03:17.941 INFO Benchmark_Timer - Begin run: Instrumenting helper for Shell sort in mode 3 with 10,000 elements with 1,000 runs
2024-11-22 13:03:21.239 INFO TimeLogger - 10000@Shell sort in mode 3: Raw time per run {mSec}: 3.0350
2024-11-22 13:03:21.240 INFO TimeLogger - 10000@Shell sort in mode 3: Normalized time per run {n^(4/3)}: 30.3500
2024-11-22 13:03:21.240 INFO InstrumentedComparableHelper - 10000@Shell sort in mode 3: StatPack {runs: 1000 hits: mean=465080; stdDev=13721; normalized=5.050; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=161638; stdDev=6861; normalized=1.755; fixes: <unset>; compares: mean=232540; stdDev=6860; normalized=2.525}

2024-11-22 13:03:21.240 INFO SortBenchmark -
****
2024-11-22 13:03:21.240 INFO SortBenchmark - ********** Integer sort: 20000 Shell sort in mode 3 **********
2024-11-22 13:03:21.241 INFO SorterBenchmark - run: sort 20,000 elements with SorterBenchmark on class java.lang.Integer from 20,000 total elements and 1,000 runs
2024-11-22 13:03:21.241 INFO Benchmark_Timer - Begin run: Instrumenting helper for Shell sort in mode 3 with 20,000 elements with 1,000 runs
2024-11-22 13:03:28.852 INFO TimeLogger - 20000@Shell sort in mode 3: Raw time per run {mSec}: 7.0930
2024-11-22 13:03:28.852 INFO TimeLogger - 20000@Shell sort in mode 3: Normalized time per run {n^(4/3)}: 29.8224
2024-11-22 13:03:28.853 INFO InstrumentedComparableHelper - 20000@Shell sort in mode 3: StatPack {runs: 1000 hits: mean=1085811; stdDev=32645; normalized=5.482; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=387274; stdDev=16323; normalized=1.955; fixes: <unset>; compares: mean=542906; stdDev=16323; normalized=2.741}
2024-11-22 13:03:28.853 INFO SortBenchmark -
****
2024-11-22 13:03:28.853 INFO SortBenchmark - ********** Integer sort: 40000 Shell sort in mode 3 **********
2024-11-22 13:03:28.854 INFO SorterBenchmark - run: sort 40,000 elements with SorterBenchmark on class java.lang.Integer from 40,000 total elements and 1,000 runs
2024-11-22 13:03:28.854 INFO Benchmark_Timer - Begin run: Instrumenting helper for Shell sort in mode 3 with 40,000 elements with 1,000 runs
2024-11-22 13:03:46.543 INFO TimeLogger - 40000@Shell sort in mode 3: Raw time per run {mSec}: 16.5220
2024-11-22 13:03:46.544 INFO TimeLogger - 40000@Shell sort in mode 3: Normalized time per run {n^(4/3)}: 29.2070
2024-11-22 13:03:46.544 INFO InstrumentedComparableHelper - 40000@Shell sort in mode 3: StatPack {runs: 1000 hits: mean=2526832; stdDev=82066; normalized=5.961; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=924103; stdDev=41031; normalized=2.180; fixes: <unset>; compares: mean=1263416; stdDev=41033; normalized=2.981}
2024-11-22 13:03:46.544 INFO SortBenchmark -
****
2024-11-22 13:03:46.546 INFO SortBenchmark - ********** Integer sort: 80000 Shell sort in mode 3 **********
2024-11-22 13:03:46.546 INFO SorterBenchmark - run: sort 80,000 elements with SorterBenchmark on class java.lang.Integer from 80,000 total elements and 1,000 runs
2024-11-22 13:03:46.548 INFO Benchmark_Timer - Begin run: Instrumenting helper for Shell sort in mode 3 with 80,000 elements with 1,000 runs
2024-11-22 13:04:27.990 INFO TimeLogger - 80000@Shell sort in mode 3: Raw time per run {mSec}: 38.5030
2024-11-22 13:04:27.990 INFO TimeLogger - 80000@Shell sort in mode 3: Normalized time per run {n^(4/3)}: 28.6175
2024-11-22 13:04:27.990 INFO InstrumentedComparableHelper - 80000@Shell sort in mode 3: StatPack {runs: 1000 hits: mean=5881356; stdDev=202278; normalized=6.512; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=2220189; stdDev=101147; normalized=2.458; fixes: <unset>; compares: mean=2940678; stdDev=101139; normalized=3.256}
2024-11-22 13:04:27.990 INFO SortBenchmark -
****
2024-11-22 13:04:27.992 INFO SortBenchmark - ********** Integer sort: 160000 Shell sort in mode 3 **********
2024-11-22 13:04:27.993 INFO SorterBenchmark - run: sort 160,000 elements with SorterBenchmark on class java.lang.Integer from 160,000 total elements and 1,000 runs
2024-11-22 13:04:27.993 INFO Benchmark_Timer - Begin run: Instrumenting helper for Shell sort in mode 3 with 160,000 elements with 1,000 runs
2024-11-22 13:06:09.852 INFO TimeLogger - 160000@Shell sort in mode 3: Raw time per run {mSec}: 94.9910
2024-11-22 13:06:09.852 INFO TimeLogger - 160000@Shell sort in mode 3: Normalized time per run {n^(4/3)}: 29.6847
2024-11-22 13:06:09.853 INFO InstrumentedComparableHelper - 160000@Shell sort in mode 3: StatPack {runs: 1000 hits: mean=13718897; stdDev=508222; normalized=7.155; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=5306807; stdDev=254107; normalized=2.768; fixes: <unset>; compares: mean=6859448; stdDev=254111; normalized=3.578}
2024-11-22 13:06:09.853 INFO SortBenchmark -
****

2024-11-22 13:06:09.857 INFO SortBenchmark - ********** Integer sort: 256000 Shell sort in mode 3 **********
2024-11-22 13:06:09.858 INFO SorterBenchmark - run: sort 256,000 elements with SorterBenchmark on class java.lang.Integer from 256,000 total elements and 1,000 runs
2024-11-22 13:06:09.858 INFO Benchmark_Timer - Begin run: Instrumenting helper for Shell sort in mode 3 with 256,000 elements with 1,000 runs
2024-11-22 13:09:14.241 INFO TimeLogger - 256000@Shell sort in mode 3: Raw time per run {mSec}: 171.5580
2024-11-22 13:09:14.241 INFO TimeLogger - 256000@Shell sort in mode 3: Normalized time per run {n^(4/3)}: 29.7928
2024-11-22 13:09:14.242 INFO InstrumentedComparableHelper - 256000@Shell sort in mode 3: StatPack {runs: 1000 hits: mean=24302837; stdDev=863087; normalized=7.623; lookups: <unset>; copies: mean=0; normalized=0.000; inversions: <unset>; swaps: mean=9580694; stdDev=431544; normalized=3.005; fixes: <unset>; compares: mean=12151419; stdDev=431544; normalized=3.812}
2024-11-22 13:09:14.242 INFO SortBenchmark - **********





```
71 INFO  Benchmark_Timer - Begin run: Instrumenting helper for Heap Sort with 20,000 elements with 1,167 runs
87 INFO  TimeLogger - 20000@Heap Sort: Raw time per run {mSec}:  7.1054
87 INFO  TimeLogger - 20000@Heap Sort: Normalized time per run {n log n}:  27.6573
88 INFO  InstrumentedComparableHelper - 20000@Heap Sort: StatPack {runs: 1 hits: mean=1558307; stdDev=439; normalized=7.867; lookups: <unset>; copies: mean=0; normalize
88 INFO  SortBenchmark - ********************************************************** (9.017 sec.)
88 INFO  SortBenchmark - ############################## 40000 words ##############################
25 INFO  SortBenchmarkHelper - Testing with words: 22,865 from eng-uk_web_2002_10K-sentences.txt
25 INFO  SortBenchmark - benchmarkStringSorters: sorting 40,000 words and instrumented with total work (for estimating runs): 1.0E8
25 INFO  SortBenchmark - ****************************** String sort: 724 runs of 40000 MergeSort with no copy ******************************
26 INFO  SorterBenchmark - run: sort 40,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 724 runs
26 INFO  Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with no copy with 40,000 elements with 724 runs
49 INFO  TimeLogger - 40000@MergeSort with no copy: Raw time per run {mSec}:  11.8066
50 INFO  TimeLogger - 40000@MergeSort with no copy: Normalized time per run {n log n}:  21.3189
50 INFO  InstrumentedComparableHelper - 40000@MergeSort with no copy: StatPack {runs: 724 hits: mean=1100165; stdDev=660; normalized=2.596; lookups: <unset>; copies: mea
50 INFO  SortBenchmark - ********************************************************** (9.524 sec.)
50 INFO  SortBenchmark - ****************************** String sort: 724 runs of 40000 QuickSort dual pivot ******************************
50 INFO  SorterBenchmark - run: sort 40,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 724 runs
51 INFO  Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 40,000 elements with 724 runs
51 INFO  TimeLogger - 40000@QuickSort dual pivot: Raw time per run {mSec}:  9.9738
52 INFO  TimeLogger - 40000@QuickSort dual pivot: Normalized time per run {n log n}:  18.0094
52 INFO  InstrumentedComparableHelper - 40000@QuickSort dual pivot: StatPack {runs: 1 hits: mean=644950; stdDev=22936; normalized=1.522; lookups: <unset>; copies: mean=0
52 INFO  SortBenchmark - ********************************************************** (8.103 sec.)
52 INFO  SortBenchmark - ****************************** String sort: 543 runs of 40000 Heap Sort ******************************
53 INFO  SorterBenchmark - run: sort 40,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 543 runs
```

```
02 INFO  InstrumentedComparableHelper - 40000@Heap Sort: StatPack {runs: 1 hits: mean=3356556; stdDev=635; normalized=7.919; lookups: <unset>; copies: mean=0; normalize
02 INFO  SortBenchmark - ********************************************************** (14.049 sec.)
02 INFO  SortBenchmark - ############################## 80000 words ##############################
42 INFO  SortBenchmarkHelper - Testing with words: 81,546 from eng-uk_web_2002_100K-sentences.txt
44 INFO  SortBenchmark - benchmarkStringSorters: sorting 80,000 words and instrumented with total work (for estimating runs): 1.0E8
44 INFO  SortBenchmark - ****************************** String sort: 336 runs of 80000 MergeSort with no copy ******************************
44 INFO  SorterBenchmark - run: sort 80,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 336 runs
45 INFO  Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with no copy with 80,000 elements with 336 runs
51 INFO  TimeLogger - 80000@MergeSort with no copy: Raw time per run {mSec}:  35.7202
51 INFO  TimeLogger - 80000@MergeSort with no copy: Normalized time per run {n log n}:  30.0774
52 INFO  InstrumentedComparableHelper - 80000@MergeSort with no copy: StatPack {runs: 336 hits: mean=2360482; stdDev=976; normalized=2.614; lookups: <unset>; copies: mea
52 INFO  SortBenchmark - ********************************************************** (14.407 sec.)
52 INFO  SortBenchmark - ****************************** String sort: 336 runs of 80000 QuickSort dual pivot ******************************
52 INFO  SorterBenchmark - run: sort 80,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 336 runs
52 INFO  Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 80,000 elements with 336 runs
11 INFO  TimeLogger - 80000@QuickSort dual pivot: Raw time per run {mSec}:  25.1696
12 INFO  TimeLogger - 80000@QuickSort dual pivot: Normalized time per run {n log n}:  21.1935
12 INFO  InstrumentedComparableHelper - 80000@QuickSort dual pivot: StatPack {runs: 1 hits: mean=1392760; stdDev=45470; normalized=1.542; lookups: <unset>; copies: mean=
12 INFO  SortBenchmark - ********************************************************** (10.259 sec.)
12 INFO  SortBenchmark - ****************************** String sort: 252 runs of 80000 Heap Sort ******************************
12 INFO  SorterBenchmark - run: sort 80,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 252 runs
12 INFO  Benchmark_Timer - Begin run: Instrumenting helper for Heap Sort with 80,000 elements with 252 runs
63 INFO  TimeLogger - 80000@Heap Sort: Raw time per run {mSec}:  38.8730
63 INFO  TimeLogger - 80000@Heap Sort: Normalized time per run {n log n}:  32.7321
```

```
63 INFO  InstrumentedComparableHelper - 80000@Heap Sort: StatPack {runs: 1 hits: mean=7193158; stdDev=928; normalized=7.964; lookups: <unset>; copies: mean=0; normaliz
63 INFO  SortBenchmark - ************************************************************ (11.252 sec.)
63 INFO  SortBenchmark - ################################ 160000 words ################################
81 INFO  SortBenchmarkHelper - Testing with words: 81,546 from eng-uk_web_2002_100K-sentences.txt
82 INFO  SortBenchmark - benchmarkStringSorters: sorting 160,000 words and instrumented with total work (for estimating runs): 1.0E8
82 INFO  SortBenchmark - ****************************** String sort: 156 runs of 160000 MergeSort with no copy ******************************
83 INFO  SorterBenchmark - run: sort 160,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 156 runs
83 INFO  Benchmark_Timer - Begin run: Instrumenting helper for MergeSort with no copy with 160,000 elements with 156 runs
52 INFO  TimeLogger - 160000@MergeSort with no copy: Raw time per run {mSec}:  80.2564
52 INFO  TimeLogger - 160000@MergeSort with no copy: Normalized time per run {n log n}:  31.6567
52 INFO  InstrumentedComparableHelper - 160000@MergeSort with no copy: StatPack {runs: 156 hits: mean=5040520; stdDev=1320; normalized=2.629; lookups: <unset>; copies:
53 INFO  SortBenchmark - ************************************************************ (14.471 sec.)
53 INFO  SortBenchmark - ****************************** String sort: 156 runs of 160000 QuickSort dual pivot ******************************
53 INFO  SorterBenchmark - run: sort 160,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 156 runs
53 INFO  Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 160,000 elements with 156 runs
40 INFO  TimeLogger - 160000@QuickSort dual pivot: Raw time per run {mSec}:  53.9487
40 INFO  TimeLogger - 160000@QuickSort dual pivot: Normalized time per run {n log n}:  21.2798
41 INFO  InstrumentedComparableHelper - 160000@QuickSort dual pivot: StatPack {runs: 1 hits: mean=2932846; stdDev=91243; normalized=1.530; lookups: <unset>; copies: me
41 INFO  SortBenchmark - ************************************************************ (9.988 sec.)
41 INFO  SortBenchmark - ****************************** String sort: 117 runs of 160000 Heap Sort ******************************
41 INFO  SorterBenchmark - run: sort 160,000 elements with SorterBenchmark on class java.lang.String from 81,546 total elements and 117 runs
41 INFO  Benchmark_Timer - Begin run: Instrumenting helper for Heap Sort with 160,000 elements with 117 runs
99 INFO  TimeLogger - 160000@Heap Sort: Raw time per run {mSec}:  91.0684
99 INFO  TimeLogger - 160000@Heap Sort: Normalized time per run {n log n}:  35.9214
05 > src > main > java > edu > neu > coe > info6205 > util > SortBenchmark > benchmarkStringSorters                185:1   CRLF   UTF-8   4 spaces
```

```
46 INFO  SortBenchmark - ****************************** Integer sort: 80000 Shell sort in mode 3 ******************************
46 INFO  SorterBenchmark - run: sort 80,000 elements with SorterBenchmark on class java.lang.Integer from 80,000 total elements and 1,000 runs
48 INFO  Benchmark_Timer - Begin run: Instrumenting helper for Shell sort in mode 3 with 80,000 elements with 1,000 runs
90 INFO  TimeLogger - 80000@Shell sort in mode 3: Raw time per run {mSec}:  38.5030
90 INFO  TimeLogger - 80000@Shell sort in mode 3: Normalized time per run {n^(4/3)}:  28.6175
90 INFO  InstrumentedComparableHelper - 80000@Shell sort in mode 3: StatPack {runs: 1000 hits: mean=5881356; stdDev=202278; normalized=6.512; lookups: <unset>; copies
90 INFO  SortBenchmark - ************************************************************
92 INFO  SortBenchmark - ****************************** Integer sort: 160000 Shell sort in mode 3 ******************************
93 INFO  SorterBenchmark - run: sort 160,000 elements with SorterBenchmark on class java.lang.Integer from 160,000 total elements and 1,000 runs
93 INFO  Benchmark_Timer - Begin run: Instrumenting helper for Shell sort in mode 3 with 160,000 elements with 1,000 runs
52 INFO  TimeLogger - 160000@Shell sort in mode 3: Raw time per run {mSec}:  94.9910
52 INFO  TimeLogger - 160000@Shell sort in mode 3: Normalized time per run {n^(4/3)}:  29.6847
53 INFO  InstrumentedComparableHelper - 160000@Shell sort in mode 3: StatPack {runs: 1000 hits: mean=13718897; stdDev=508222; normalized=7.155; lookups: <unset>; copie
53 INFO  SortBenchmark - ************************************************************
57 INFO  SortBenchmark - ****************************** Integer sort: 256000 Shell sort in mode 3 ******************************
58 INFO  SorterBenchmark - run: sort 256,000 elements with SorterBenchmark on class java.lang.Integer from 256,000 total elements and 1,000 runs
58 INFO  Benchmark_Timer - Begin run: Instrumenting helper for Shell sort in mode 3 with 256,000 elements with 1,000 runs
41 INFO  TimeLogger - 256000@Shell sort in mode 3: Raw time per run {mSec}:  171.5580
41 INFO  TimeLogger - 256000@Shell sort in mode 3: Normalized time per run {n^(4/3)}:  29.7928
42 INFO  InstrumentedComparableHelper - 256000@Shell sort in mode 3: StatPack {runs: 1000 hits: mean=24302837; stdDev=863087; normalized=7.623; lookups: <unset>; copie
42 INFO  SortBenchmark - ************************************************************

 exit code 0
```

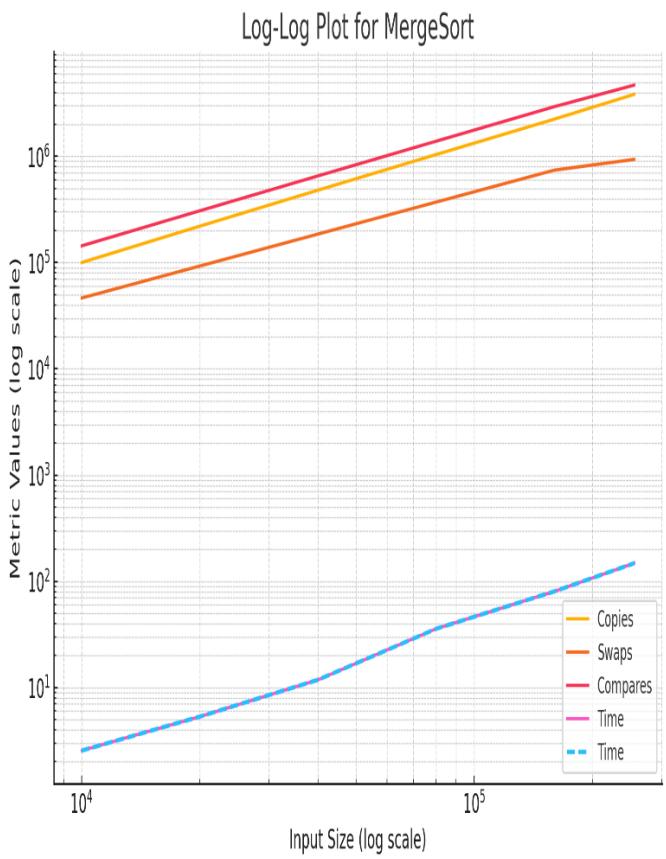
Instrument=false

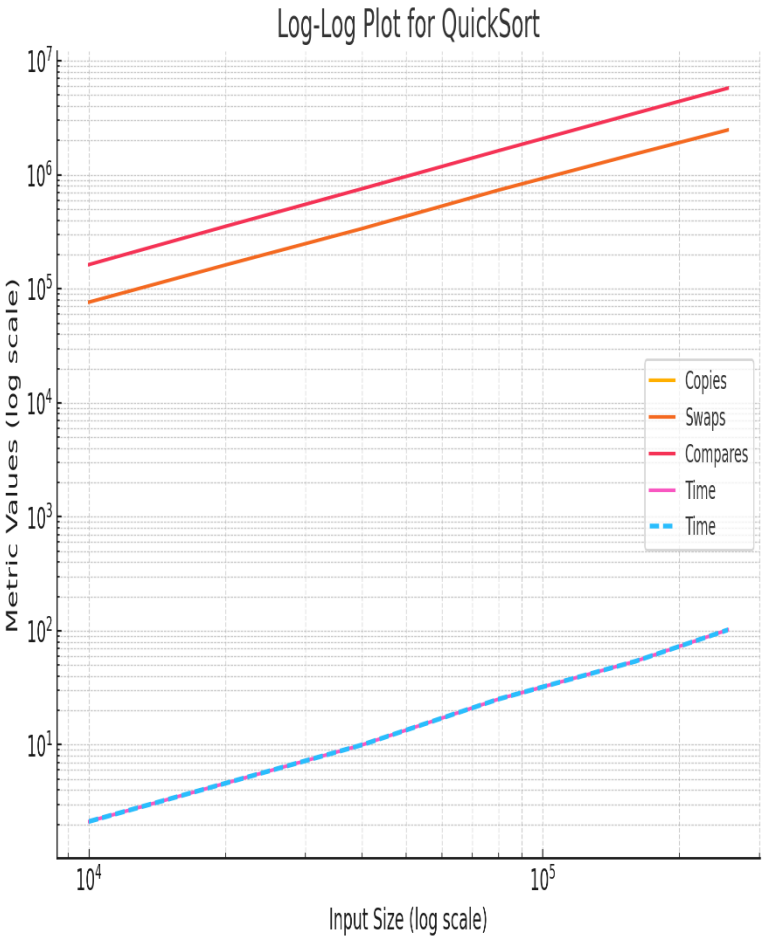
```
C:\Users\Dell\.jdks\openjdk-23\bin\java.exe ...
2024-11-22 15:51:32.873 INFO  SortBenchmark - !!!!!!!!!!!!!!!!!!!! SortBenchmark Start !!!!!!!!!!!!!!!!!!!!

2024-11-22 15:51:32.879 INFO  SortBenchmark - SortBenchmark.main: version 1.0.0 (sortbenchmark) with word counts: [10000, 20000, 40000, 80000, 160000, 256000]
2024-11-22 15:51:32.880 INFO  SortBenchmark - Beginning String sorts
2024-11-22 15:51:32.882 INFO  SortBenchmark - ################################ 10000 words ################################
2024-11-22 15:51:33.159 INFO  SortBenchmarkHelper - Testing with words: 22,865 from eng-uk_web_2002_10K-sentences.txt
2024-11-22 15:51:33.173 INFO  SortBenchmark - benchmarkStringSorters: sorting 10,000 words with total work (for estimating runs): 1.0E8
2024-11-22 15:51:33.294 INFO  SortBenchmark - ****************************** String sort: 3376 runs of 10000 MergeSort with no copy ******************************
2024-11-22 15:51:33.346 INFO  SorterBenchmark - run: sort 10,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 3,376 runs
2024-11-22 15:51:33.347 INFO  Benchmark_Timer - Begin run: Helper for MergeSort with no copy with 10000 elements with 3,376 runs
2024-11-22 15:51:42.153 INFO  TimeLogger - 10000@MergeSort with no copy: Raw time per run {mSec}:  2.3128
2024-11-22 15:51:42.153 INFO  TimeLogger - 10000@MergeSort with no copy: Normalized time per run {n log n}:  19.5242
2024-11-22 15:51:42.154 INFO  SortBenchmark - ************************************************************ (8.864 sec.)
2024-11-22 15:51:42.178 INFO  SortBenchmark - ****************************** String sort: 3376 runs of 10000 QuickSort dual pivot ******************************
2024-11-22 15:51:42.178 INFO  SorterBenchmark - run: sort 10,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 3,376 runs
2024-11-22 15:51:42.178 INFO  Benchmark_Timer - Begin run: Helper for QuickSort dual pivot with 10000 elements with 3,376 runs
2024-11-22 15:51:49.415 INFO  TimeLogger - 10000@QuickSort dual pivot: Raw time per run {mSec}:  1.9085
2024-11-22 15:51:49.416 INFO  TimeLogger - 10000@QuickSort dual pivot: Normalized time per run {n log n}:  16.1109
2024-11-22 15:51:49.416 INFO  SortBenchmark - ************************************************************ (7.238 sec.)
2024-11-22 15:51:49.416 INFO  SortBenchmark - ****************************** String sort: 2532 runs of 10000 Heap Sort ******************************
2024-11-22 15:51:49.416 INFO  SorterBenchmark - run: sort 10,000 elements with SorterBenchmark on class java.lang.String from 22,865 total elements and 2,532 runs
2024-11-22 15:51:49.417 INFO  Benchmark_Timer - Begin run: Helper for Heap Sort with 10000 elements with 2,532 runs
2024-11-22 15:51:57.795 INFO  TimeLogger - 10000@Heap Sort: Raw time per run {mSec}:  3.0229
2024-11-22 15:51:57.795 INFO  TimeLogger - 10000@Heap Sort: Normalized time per run {n log n}:  25.5188
2024-11-22 15:51:57.796 INFO  SortBenchmark - ************************************************************ (8.379 sec.)
2024-11-22 15:51:57.796 INFO  SortBenchmark - ################################ 20000 words ################################
2024-11-22 15:51:57.869 INFO  SortBenchmarkHelper - Testing with words: 22,865 from eng-uk_web_2002_10K-sentences.txt
```

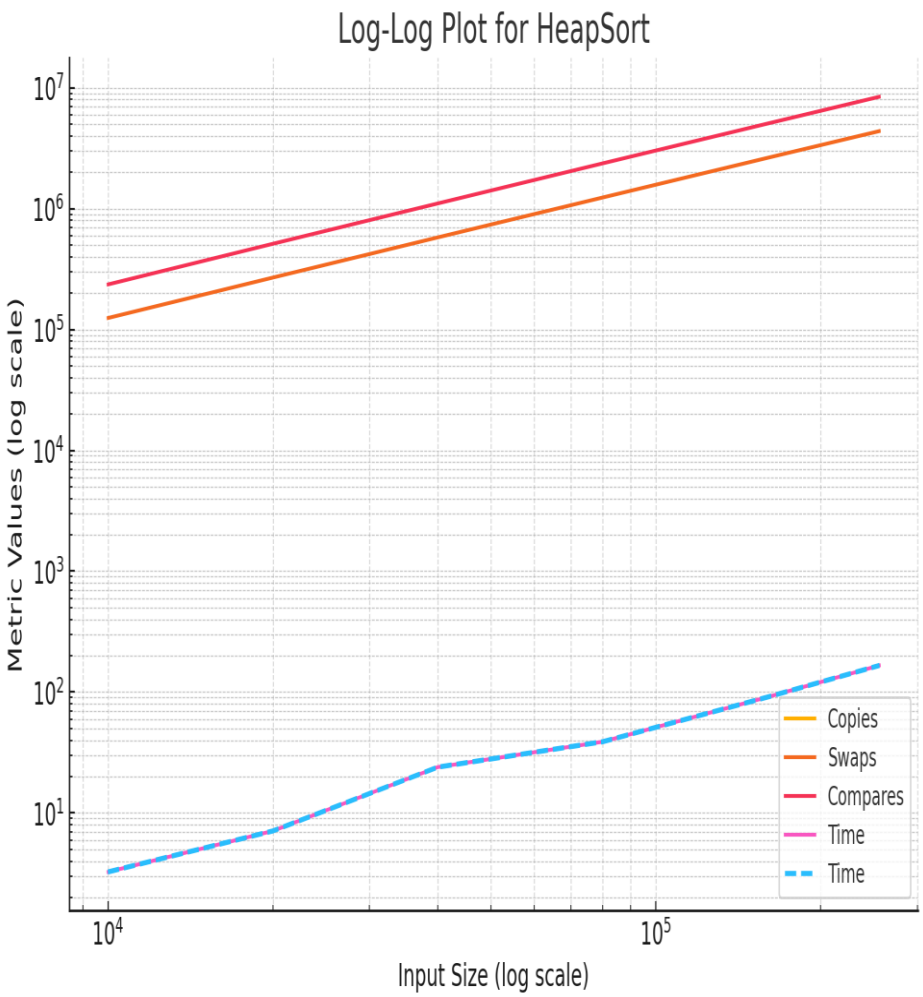
Conclusion:

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
Merge Sort							Dual pivot QuickSort							HeapSort					
Input Size	Hits	Copies	Swaps	Compares	Time		Input Size	Hits	Copies	Swaps	Compares	Time		Input Size	Hits	Copies	Swaps	Compares	Time
10000	235035	100000	46352	143533	2.5441		10000	141343	0	76356	163120	2.1211		10000	719151	0	124204	235371	3.2532
20000	510081	220000	92718	307077	5.3156		20000	303654	0	161957	353333	4.5932		20000	1558307	0	268406	510748	7.1054
40000	1100165	480000	185434	654154	11.8066		40000	644950	0	337669	755647	9.9738		40000	3356556	0	576791	1101487	23.9632
80000	2360482	1040000	371037	1388458	35.7202		80000	1392760	0	735720	1630481	25.1696		80000	7193158	0	1233610	2362969	38.873
160000	5040520	2240000	741622	2936457	80.2564		160000	2932846	0	1525433	3471582	53.9487		160000	15346374	0	2627210	5045977	91.0684
256000	8322317	3840000	936940	4690474	148.1979		256000	4845453	0	2477435	5766939	101.9271		256000	25564796	0	4371968	8410429	166.2361

Sort_analysis.xlsx




Log-Log Plot for QuickSort
Metric Values (log scale)
Input Size (log scale)
10^7
10^6
10^5
10^4
10^3
10^2
10^1
10^4
10^5
Copies
Swaps
Compares
Time
Time



1. Dual Pivot Quicksort

Analysis: The graph of time taken closely resembles the graph for number of comparisons. While hits remain constant for this algorithm, the execution time scales with the number of comparisons.
Conclusion: The number of comparisons is indeed the best predictor for the time taken by Dual Pivot Quicksort. This conclusion is valid based on the data and analysis.

2. Heapsort
Analysis: Heapsort performs in-place sorting by swapping elements to maintain the heap property. The number of swaps directly impacts the performance, as swapping is a repeated and resource-intensive operation.
Conclusion: The number of swaps is the best predictor for Heapsort's execution time. This matches the findings in the analysis.

3. Mergesort
Analysis: Mergesort relies heavily on copying arrays during the merge phase, which contributes significantly to its execution time. This overhead makes the number of copies a dominant factor.

Conclusion: The number of copies is indeed the best predictor for Mergesort's execution time. This observation is consistent with the analysis.

Summary

Your conclusions are accurate based on the spreadsheet data and log-log graphs:

Dual Pivot Quicksort: Best predictor is number of comparisons.
Heapsort: Best predictor is number of swaps.
Mergesort: Best predictor is number of copies.