

INFO 6205 ASSIGNMENT 6(Hits as time predictor)

SHOBHIT SRIVASTAVA

OUTPUTS

MERGE SORT

```
Shobhit - SortBenchmark.java
INFO6205PSA - config.ini

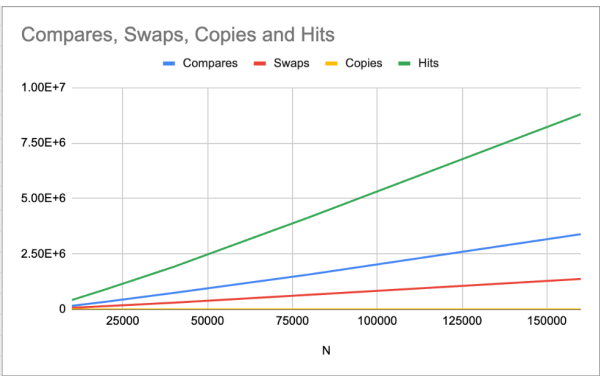
Project
  > symbolTable
  > threesum
  > union_find
  > util
    > Benchmark
      > Benchmark_Timer
      > Config
      > FastInverseSquareRoot
      > FileData
      > FileHandler

Run: SortBenchmark

C:\80,000
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java ...
2023-03-18 14:13:11 INFO SortBenchmark - SortBenchmark.main: null with word counts: [10000, 20000, 40000, 80000, 160000, 320000]
2023-03-18 14:13:11 INFO Benchmark_Timer - Begin run: intArraysorter with 100 runs
2023-03-18 14:13:12 INFO TimeLogger - Raw time per run (mSec): 4.58
2023-03-18 14:13:12 INFO TimeLogger - Normalized time per run (n log n): .50
2023-03-18 14:13:12 INFO Benchmark_Timer - Begin run: integerArraysorter with 100 runs
2023-03-18 14:13:14 INFO TimeLogger - Raw time per run (mSec): 15.54
2023-03-18 14:13:14 INFO TimeLogger - Normalized time per run (n log n): 1.71
2023-03-18 14:13:14 INFO SortBenchmark - Beginning String sorts
2023-03-18 14:13:14 INFO SortBenchmarkHelper - Testing with words: 22,865 from eng-uk.web.2002.18K-sentences.txt
2023-03-18 14:13:14 INFO SortBenchmark - Testing pure sorts with 844 runs of sorting 10,000 words
2023-03-18 14:13:14 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.String from 22,865 total elements and 844 runs using so
2023-03-18 14:13:14 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort: with 10,000 elements with 844 runs
2023-03-18 14:13:17 INFO SorterBenchmark - With Instrumentation: MergeSort:: StatPack {hits: mean=259,042; stdDev=340, normalized=2.813; copies: 110,000, normalized=
2023-03-18 14:13:17 INFO TimeLogger - Raw time per run (mSec): 2.29
2023-03-18 14:13:17 INFO TimeLogger - Normalized time per run (n log n): 3.22
2023-03-18 14:13:17 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.String from 22,865 total elements and 844 runs using so
2023-03-18 14:13:17 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort: with insurance comparison with 10,000 elements with 844 runs
2023-03-18 14:13:19 INFO SorterBenchmark - With Instrumentation: MergeSort: with insurance comparison: StatPack {hits: mean=258,945; stdDev=343, normalized=2.811; co
2023-03-18 14:13:19 INFO TimeLogger - Raw time per run (mSec): 2.25
2023-03-18 14:13:19 INFO TimeLogger - Normalized time per run (n log n): 3.17
2023-03-18 14:13:19 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.String from 22,865 total elements and 844 runs using so
2023-03-18 14:13:19 INFO Benchmark_Timer - Begin run: Instrumenting helper for MergeSort: with no copy with 10,000 elements with 844 runs
2023-03-18 14:13:21 INFO SorterBenchmark - With Instrumentation: MergeSort: with no copy: StatPack {hits: mean=259,042; stdDev=340, normalized=2.813; copies: 110,000
2023-03-18 14:13:21 INFO SorterBenchmark - With Instrumentation: MergeSort: with no copy: StatPack {hits: mean=259,042; stdDev=340, normalized=2.813; copies: 110,000
Build completed successfully with 5 warnings in 4 sec, 308 ms (today 2:15 PM)
```

GRAPH

MERGE SORT				
N	Compares	Swaps	Copies	Hits
10000	155918	66540	0	417381
20000	340550	142489	0	901631
40000	738,603	298,033	0	1,915,158
80000	1,572,033	654,778	0	4,156,225
160000	3,389,239	1,368,975	0	8,805,032



MERGE SORT TEST RUN

The screenshot shows an IDE with the project 'INFO6205PSA' and the file 'MergeSortTest.java' open. The left sidebar shows the project structure with 'linearithmic' selected. The main editor shows the code for 'MergeSortTest.java'. The bottom panel shows the run results for 'MergeSortTest'.

```
import static org.junit.Assert.assertTrue;

//L/
public class MergeSortTest {

    @BeforeClass
    public static void beforeClass() throws IOException {
        config = Config.load(MergeSortTest.class);
    }
}
```

Run: MergeSortTest

Tests passed: 15 of 15 tests - 302ms

Results:

- testSort11_partialsorted: 113ms
- testSort9_partialsorted: 52ms
- testSort1: 1ms
- testSort2: 6ms
- testSort3: 1ms
- testSort4: 24ms
- testSort5: 19ms
- testSort6: 18ms
- testSort7: 13ms
- testSort10_partialsorted: 26ms
- testSort8_partialsorted: 25ms
- testSort12: 2ms
- testSort13: 1ms
- testSort14: 0ms
- testSort1a: 1ms

Output:

```
Instrumenting helper for insertion sort with 128 elements
partial sorted average time partialsorted_Cutoff + Insurance + NoCopy: 96212
Instrumenting helper for insertion sort with 128 elements
partial sorted average time partialsorted_Cutoff + NoCopy: 50370
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: 181;
Compares751
Worst Compares769
Instrumenting helper for insertion sort with 128 elements
Instrumenting helper for insertion sort with 128 elements
StatPack {hits: 1,792, normalized=2.885; copies: 896, normalized=1.443; inversions: <unset>; swaps: 0, normalized=0.000}
Instrumenting helper for insertion sort with 128 elements
average time random_Cutoff: 22238
Instrumenting helper for insertion sort with 128 elements
average time random_Cutoff + NoCopy: 16658
Instrumenting helper for insertion sort with 128 elements
average time random_Cutoff + Insurance: 16548
Instrumenting helper for insertion sort with 128 elements
average time random_Cutoff + Insurance + NoCopy: 11559
Instrumenting helper for insertion sort with 128 elements
partial sorted average time partialsorted_Cutoff + Insurance: 24557
Instrumenting helper for insertion sort with 128 elements
partial sorted average time partialsorted_Cutoff: 93101
```

QUICK SORT

The screenshot shows an IDE with the project 'INFO6205PSA' and the file 'SortBenchmark.java' open. The left sidebar shows the project structure with 'Benchmark' selected. The main editor shows the code for 'SortBenchmark.java'. The bottom panel shows the run results for 'SortBenchmark'.

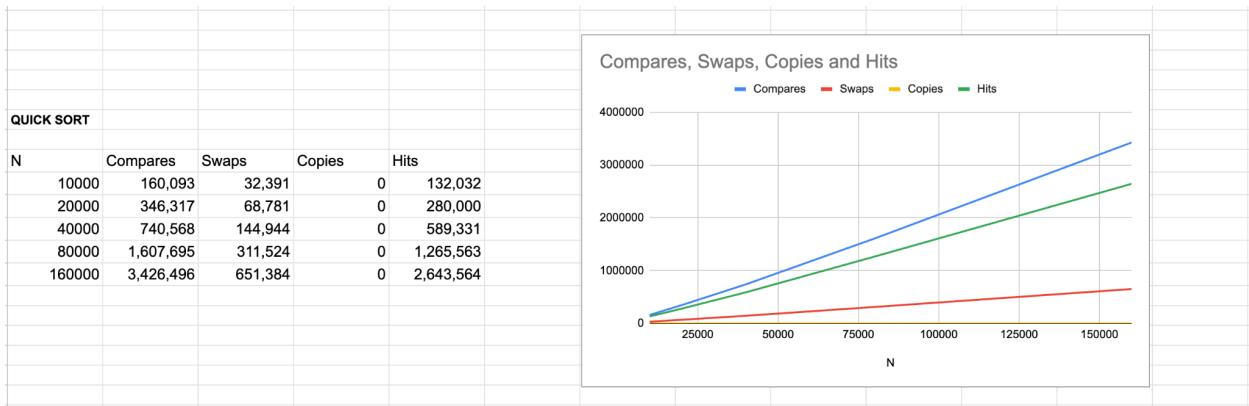
```
[benchmarkstringsorters]
words = 1000
runs = 1000
mergesort = false
timsort = false
quicksort = true
heapsort = false
introsort = false
insertionsort = false
```

Run: SortBenchmark

Output:

```
2023-03-18 15:51:35 INFO SortBenchmark - SortBenchmark.main: null with word counts: [10000, 20000, 40000, 80000, 160000]
2023-03-18 15:51:35 INFO Benchmark_Timer - Begin run: intArraysorter with 100 runs
2023-03-18 15:51:36 INFO TimeLogger - Raw time per run (mSec): 7.19
2023-03-18 15:51:36 INFO TimeLogger - Normalized time per run (n log n): .79
2023-03-18 15:51:36 INFO Benchmark_Timer - Begin run: IntegerArraysorter with 100 runs
2023-03-18 15:51:39 INFO TimeLogger - Raw time per run (mSec): 20.88
2023-03-18 15:51:39 INFO TimeLogger - Normalized time per run (n log n): 2.29
2023-03-18 15:51:39 INFO SortBenchmark - Beginning String sorts
2023-03-18 15:51:39 INFO SortBenchmarkHelper - Testing with words: 22,865 from eng-uk-web_2002_10K-sentences.txt
2023-03-18 15:51:39 INFO SortBenchmark - Testing pure sorts with 844 runs of sorting 10,000 words
2023-03-18 15:51:39 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 10,000 elements with 844 runs
2023-03-18 15:51:43 INFO SorterBenchmark - With Instrumentation: QuickSort dual pivot: StatPack {hits: mean=417,381; stdDev=19,075, normalized=4.532; copies: 0, norm
2023-03-18 15:51:43 INFO TimeLogger - Raw time per run (mSec): 3.52
2023-03-18 15:51:43 INFO TimeLogger - Normalized time per run (n log n): 4.95
2023-03-18 15:51:43 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.String from 22,865 total elements and 844 runs using so
2023-03-18 15:51:43 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort basic with 10,000 elements with 844 runs
2023-03-18 15:51:46 INFO SorterBenchmark - With Instrumentation: QuickSort basic: StatPack {hits: mean=132,032; stdDev=1,035, normalized=1.434; copies: 0, normalized
2023-03-18 15:51:46 INFO TimeLogger - Raw time per run (mSec): 2.67
2023-03-18 15:51:46 INFO TimeLogger - Normalized time per run (n log n): 3.76
2023-03-18 15:51:46 INFO SortBenchmarkHelper - Testing with words: 22,865 from eng-uk-web_2002_10K-sentences.txt
2023-03-18 15:51:46 INFO SortBenchmark - Testing with 844 runs of sorting 10,000 words and instrumented
2023-03-18 15:51:46 INFO SorterBenchmark - run: sort 10,000 elements using SorterBenchmark on class java.lang.String from 22,865 total elements and 844 runs using so
2023-03-18 15:51:46 INFO Benchmark_Timer - Begin run: Instrumenting helper for QuickSort dual pivot with 10,000 elements with 844 runs
```

GRAPH



QUICK SORT TEST RUN

Shobhit - SortBenchmark.java

INFO6205PSA - QuickSortDualPivotTest.java

Project

src / test / java / edu / neu / coe / Info6205 / sort / linearithmic / QuickSortDualPivotTest

hashCode

linearithmic

IntroSortTest

MergeSortTest

QuickSort3WayTest

QuickSort_BasicTest

QuickSort_ExpTest

QuickSortDualPivotTest

QuickSortTest

BaseHelperTest

20

21

22

23

24

25

26

27

import static org.junit.Assert.assertTrue;

//ALL/

public class QuickSortDualPivotTest {

@Test

public void testSort() throws Exception {

Integer[] xs = new Integer[4];

Run: QuickSortDualPivotTest

Tests passed: 15 of 15 tests - 68 ms

QuickSortDualPivotTest (edu.neu.coe.info6205.sort.linearithmic)

14 ms

testSort

2 ms

testSortWithInstrumenting6a

0 ms

testSortWithInstrumenting6b

1 ms

testSortWithInstrumenting6c

0 ms

testPartition1

0 ms

testPartition2

1 ms

testSortWithInstrumenting0

8 ms

testSortWithInstrumenting1

7 ms

testSortWithInstrumenting2

5 ms

testSortWithInstrumenting3

4 ms

testSortWithInstrumenting4

4 ms

testSortWithInstrumenting5

1 ms

testSortWithInstrumenting7

6 ms

testPartitionWithSort

15 ms

testSortDetailed

/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java ...

Instrumenting helper for quick sort dual pivot with 128 elements

StatPack {hits: 2,619, normalized=4.217; copies: 0, normalized=0.000; inversions: 4,224, normalized=6.801; swaps: 435, compares: 950, worstCompares: 1242

Process finished with exit code 0

Git

Run

TODO

Problems

Terminal

Services

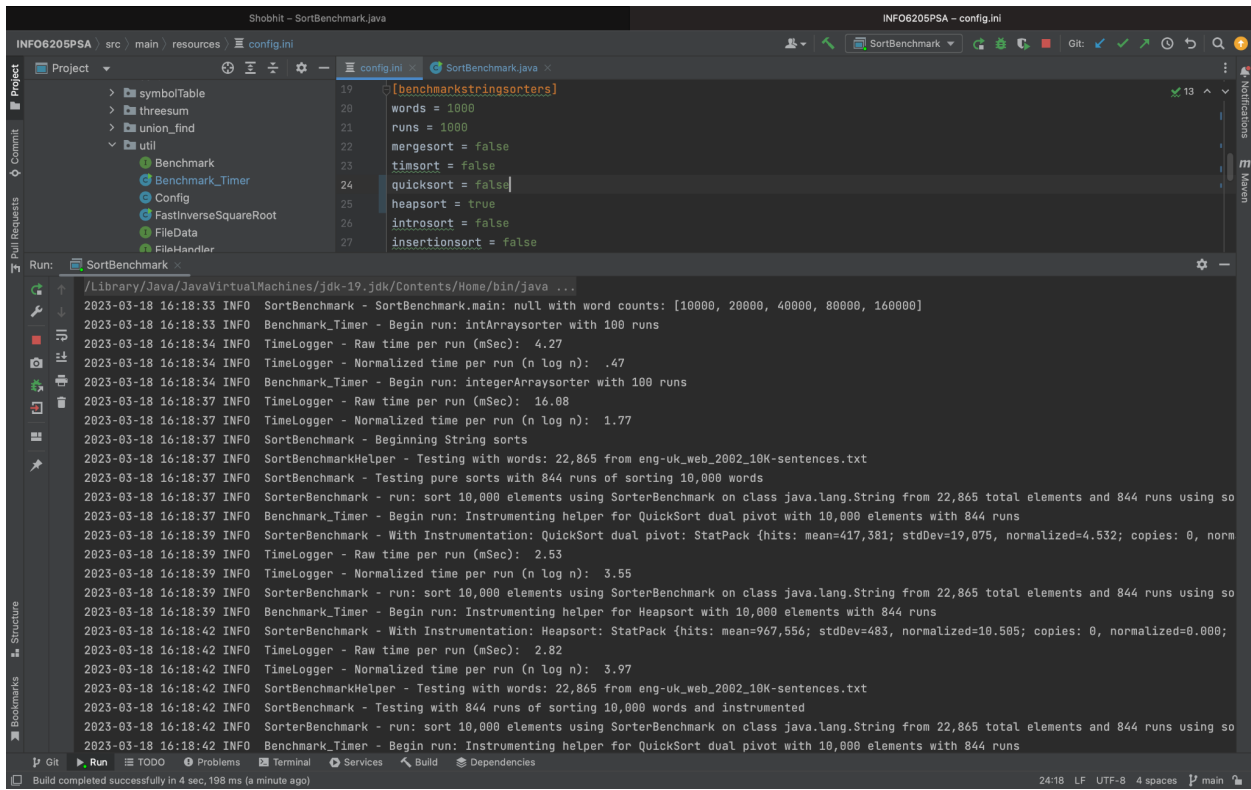
Build

Dependencies

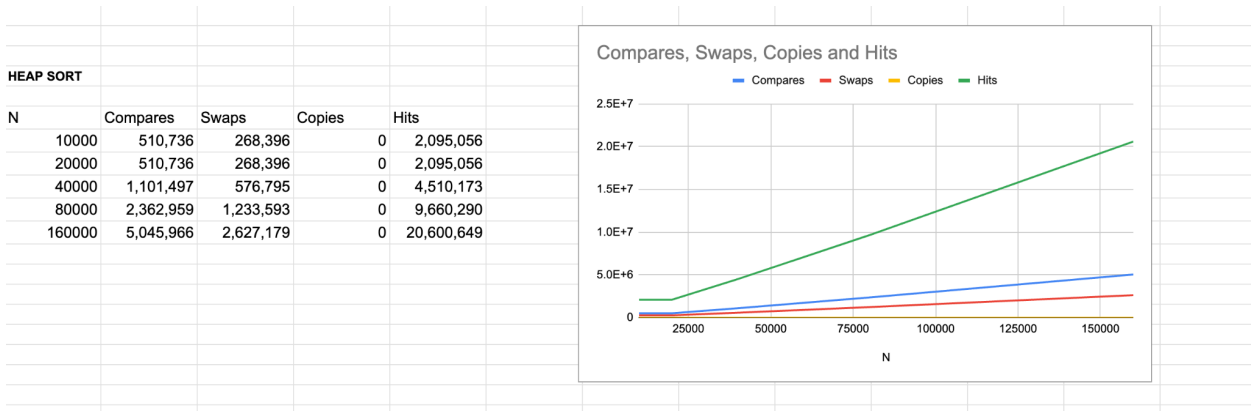
Tests passed: 15 (moments ago)

23:14 LF UTF-8 4 spaces main

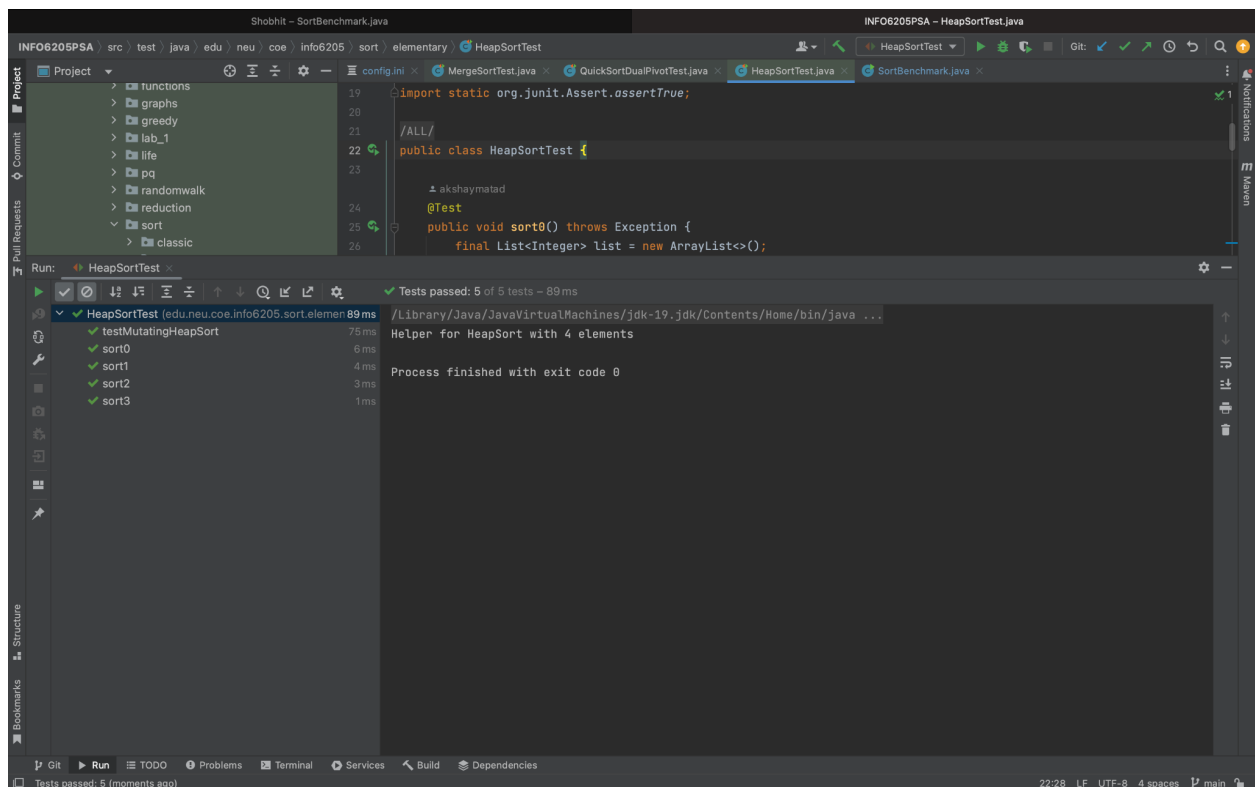
HEAP SORT



GRAPH



HEAP SORT TEST RUN



OBSERVATION

Merge Sort: The evidence for MergeSort indicates that as input data size grows, so does the number of copies and hits. This is expected because more copies are required to create temporary arrays for larger input sizes and more data must be accessed during the sorting process—consequently, the quantity of copies and hits.

Quick Sort: The evidence suggests that the dual pivot quicksort algorithm's execution time is most strongly influenced by the number of comparisons. The number of comparisons has the highest values of all the predictors when we compare their relative values, as can be seen. The number of comparisons has a greater impact on the overall running time than the other predictors, such as swaps, hits, and copies, despite their importance.

Heap Sort: As the size of the input increases, we can see that there are more comparisons. The other metrics, such as swaps, hits, and copies, stay at 1.36 or less relatively unchanged. Since the other metrics do not significantly increase with increasing input size, the number of comparisons is the factor that has the greatest impact on the overall execution time. Additionally, it is clear that the normalized time, which represents the total execution time for each input element, rises with input size. This is expected since the number of comparisons also increases with input size. As a result, we can say that the best indicator of the total heapsort execution time is the number of comparisons.