

Program Structures and Algorithms Spring 2024

NAME: Saurabh Srivastava

NUID: 002895225

GITHUB LINK: <https://github.com/ssaurabh760/INFO6205>

Task: Assignment3- Benchmark

Observations:

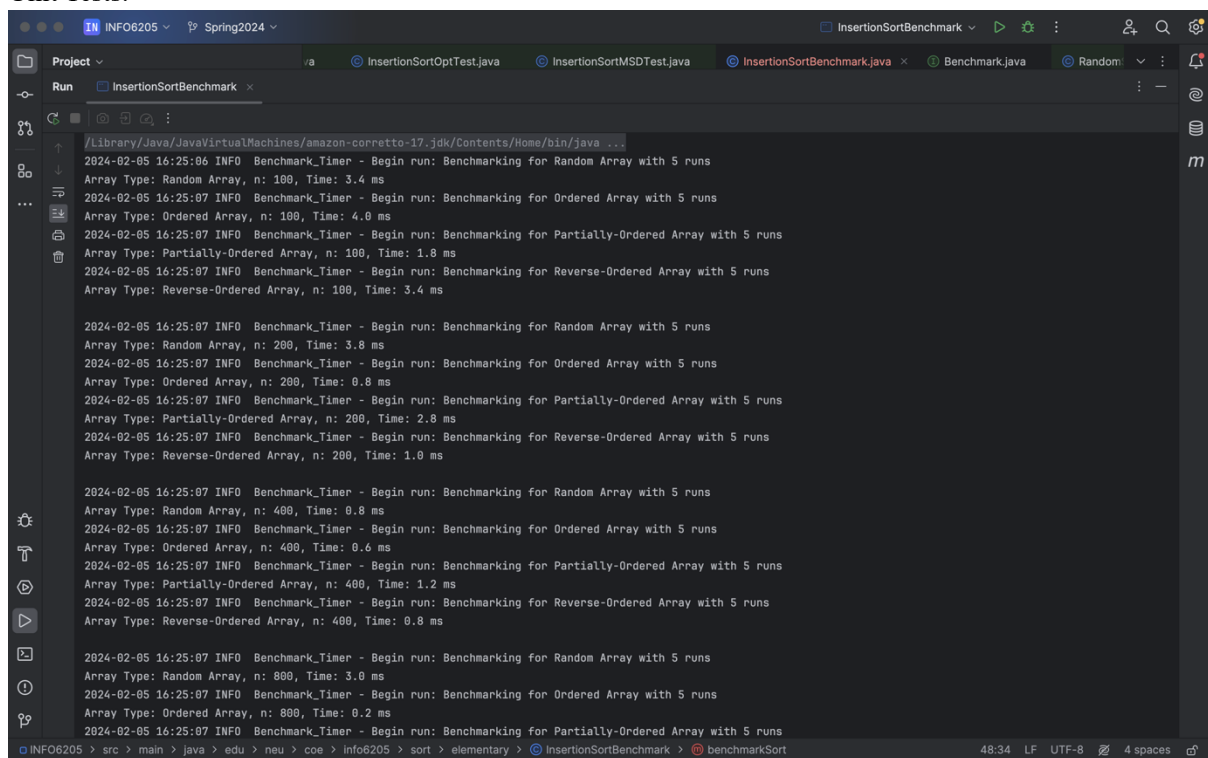
Insertion Sort works the fastest in case of sorted array because its runtime of $O(n)$ is used only for checking and since all the elements are sorted there's no swapping.

However in the case of partially and randomly sorted arrays the insertion sort can slow down the runtime complexity can be $O(n^2)$ because some of the elements might be sorted and time is not consumed in swapping the.

In case of reversed sorted the insertion sort works slowest with runtime complexity of $O(n^2)$.

So if we rank the arrays based on the time, Sorted < Partially Sorted < Randomly Sorted < Reverse Sorted.

Unit Tests:



```
INFO6205 > Spring2024 > InsertionSortBenchmark > Run
/Library/Java/JavaVirtualMachines/amazon-corretto-17-jdk/Contents/Home/bin/java ...
2024-02-05 16:25:06 INFO Benchmark_Timer - Begin run: Benchmarking for Random Array with 5 runs
Array Type: Random Array, n: 100, Time: 3.4 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Ordered Array with 5 runs
Array Type: Ordered Array, n: 100, Time: 4.0 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Partially-Ordered Array with 5 runs
Array Type: Partially-Ordered Array, n: 100, Time: 1.8 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 100, Time: 3.4 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Random Array with 5 runs
Array Type: Random Array, n: 200, Time: 3.8 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Ordered Array with 5 runs
Array Type: Ordered Array, n: 200, Time: 0.8 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Partially-Ordered Array with 5 runs
Array Type: Partially-Ordered Array, n: 200, Time: 2.8 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 200, Time: 1.0 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Random Array with 5 runs
Array Type: Random Array, n: 400, Time: 0.8 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Ordered Array with 5 runs
Array Type: Ordered Array, n: 400, Time: 0.6 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Partially-Ordered Array with 5 runs
Array Type: Partially-Ordered Array, n: 400, Time: 1.2 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 400, Time: 0.8 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Random Array with 5 runs
Array Type: Random Array, n: 800, Time: 3.0 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Ordered Array with 5 runs
Array Type: Ordered Array, n: 800, Time: 0.2 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Partially-Ordered Array with 5 runs
Array Type: Partially-Ordered Array, n: 800, Time: 1.2 ms
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 800, Time: 0.8 ms
```

INFO6205 Spring2024 InsertionSortBenchmark

```
2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 280, Time: 1.0 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Random Array with 5 runs
Array Type: Random Array, n: 400, Time: 0.8 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Ordered Array with 5 runs
Array Type: Ordered Array, n: 400, Time: 0.6 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Partially-Ordered Array with 5 runs
Array Type: Partially-Ordered Array, n: 400, Time: 1.2 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 400, Time: 0.8 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Random Array with 5 runs
Array Type: Random Array, n: 800, Time: 3.0 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Ordered Array with 5 runs
Array Type: Ordered Array, n: 800, Time: 0.2 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Partially-Ordered Array with 5 runs
Array Type: Partially-Ordered Array, n: 800, Time: 1.4 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 800, Time: 2.6 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Random Array with 5 runs
Array Type: Random Array, n: 1600, Time: 5.4 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Ordered Array with 5 runs
Array Type: Ordered Array, n: 1600, Time: 0.6 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Partially-Ordered Array with 5 runs
Array Type: Partially-Ordered Array, n: 1600, Time: 4.8 ms

2024-02-05 16:25:07 INFO Benchmark_Timer - Begin run: Benchmarking for Reverse-Ordered Array with 5 runs
Array Type: Reverse-Ordered Array, n: 1600, Time: 11.2 ms

Process finished with exit code 0
```

INFO6205 > src > main > java > edu > neu > coe > info6205 > sort > elementary > InsertionSortBenchmark > benchmarkSort 48:34 LF UTF-8 4 spaces

INFO6205 Spring2024 InsertionSortTest

```
49: nuanlin
50:
51: c void sort1() throws Exception {
52:     inal List<Integer> list = new ArrayList<>();
53:     list.add(3);
54:     list.add(4);
55:     list.add(2);
56:     list.add(1);
57:     nteger[] xs = list.toArray(new Integer[0]);
58:     aseHelper<Integer> helper = new BaseHelper<>("InsertionSort", xs.length, Config.load(InsertionSortTest.class));
59:     enericSort<Integer> sorter = new InsertionSort<>(helper);
60:     nteger[] ys = sorter.sort(xs);
61:     ssertTrue(helper.sorted(ys));
62:     ystem.out.println(sorter.toString());
63:
64:     nuanlin
65:
66: c void testMutatingInsertionSort() throws IOException {
67:     inal List<Integer> list = new ArrayList<>();
68:     list.add(3);
69: }
```

Run InsertionSortTest

InsertionSortTest (edu.nei.438 ms) Tests passed: 6 of 6 tests - 438 ms

- testMutatingInsertionSort 327 ms
- sort0 62 ms
- sort1 23 ms
- sort2 15 ms
- sort3 8 ms
- testStaticInsertionSort 3 ms

Process finished with exit code 0

INFO6205 > src > test > java > edu > neu > coe > info6205 > sort > elementary > InsertionSortTest 56:21 LF UTF-8 4 spaces

