Akshaysingh Bayes (NU ID: 002956209)

6205 - Program Structures and Algorithms

Assignment - 2

Problem Statement:

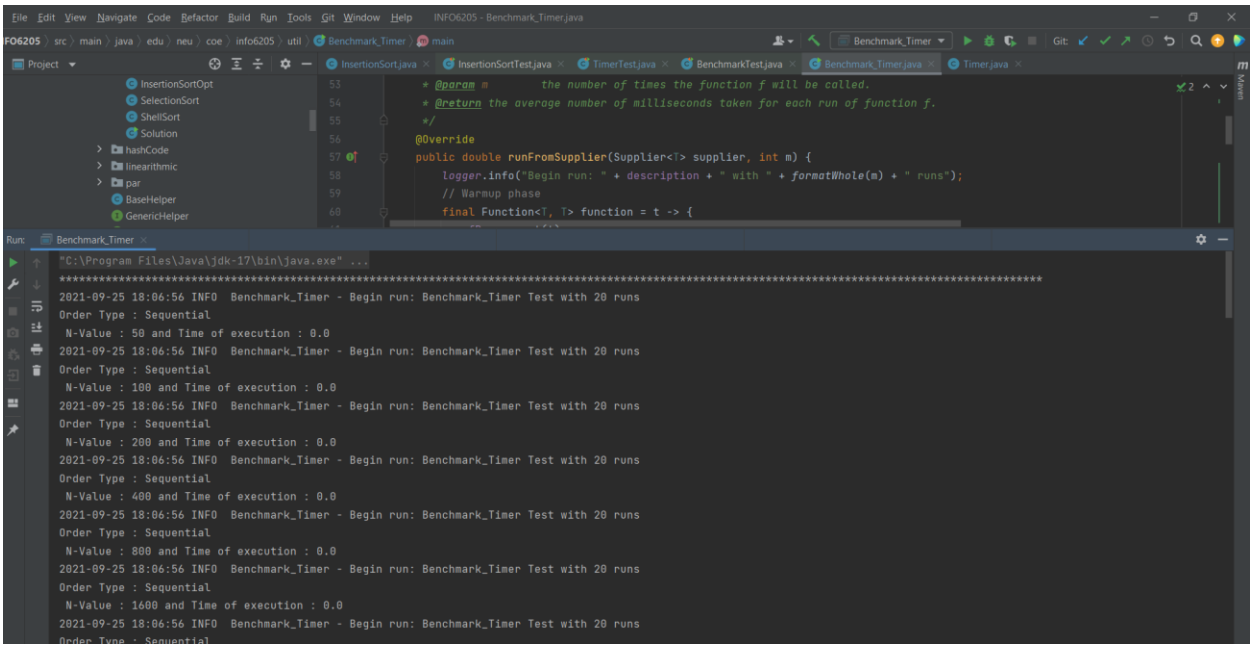
(Part 1) You are to implement three methods of a class called *Timer*

(Part 2) Implement *Insertion Sort* (in the *Insertion Sort* class) by simply looking up the insertion code used by *Arrays.sort()*.

(Part 3) Implement a main program (or you could do it via your own unit tests) to actually run the following benchmarks: measure the running times of this sort, using four different initial array ordering situations: random, ordered, partially ordered and reverse ordered.

Output:

Benchmark_Timer main () ran successfully. It is ran for 4 types of sorted arrays viz. Ordered, Partially ordered, Reversed and Random.



Console Output:

2021-09-25 18:06:56 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Sequential

N-Value : 50 and Time of execution : 0.0

2021-09-25 18:06:56 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Sequential

N-Value : 100 and Time of execution : 0.0

2021-09-25 18:06:56 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Sequential

N-Value : 200 and Time of execution : 0.0

2021-09-25 18:06:56 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Sequential

N-Value : 400 and Time of execution : 0.0

2021-09-25 18:06:56 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Sequential

N-Value : 800 and Time of execution : 0.0

2021-09-25 18:06:56 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Sequential

N-Value : 1600 and Time of execution : 0.0

2021-09-25 18:06:56 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Sequential

N-Value : 3200 and Time of execution : 0.05

2021-09-25 18:06:56 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Sequential

N-Value : 6400 and Time of execution : 0.1

2021-09-25 18:06:57 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Sequential

N-Value : 12800 and Time of execution : 0.05

2021-09-25 18:06:57 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Sequential

N-Value : 25600 and Time of execution : 0.05

****

2021-09-25 18:06:57 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Random

N-Value : 50 and Time of execution : 0.05

2021-09-25 18:06:57 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Random

N-Value : 100 and Time of execution : 0.05

2021-09-25 18:06:57 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Random

N-Value : 200 and Time of execution : 0.5

2021-09-25 18:06:57 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Random

N-Value : 400 and Time of execution : 0.25

2021-09-25 18:06:57 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Random

N-Value : 800 and Time of execution : 0.65

2021-09-25 18:06:57 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Random

N-Value : 1600 and Time of execution : 2.75

2021-09-25 18:06:57 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Random

N-Value : 3200 and Time of execution : 10.55

2021-09-25 18:06:57 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Random

N-Value : 6400 and Time of execution : 37.5

2021-09-25 18:06:58 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Random

N-Value : 12800 and Time of execution : 151.9

2021-09-25 18:07:01 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Random

N-Value : 25600 and Time of execution : 666.75

****
****

2021-09-25 18:07:16 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Partially Sequential

N-Value : 50 and Time of execution : 0.0

2021-09-25 18:07:16 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Partially Sequential

N-Value : 100 and Time of execution : 0.0

2021-09-25 18:07:16 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Partially Sequential

N-Value : 200 and Time of execution : 0.0

2021-09-25 18:07:16 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Partially Sequential

N-Value : 400 and Time of execution : 0.05

2021-09-25 18:07:16 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Partially Sequential

N-Value : 800 and Time of execution : 0.25

2021-09-25 18:07:16 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Partially Sequential

N-Value : 1600 and Time of execution : 1.4

2021-09-25 18:07:16 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Partially Sequential

N-Value : 3200 and Time of execution : 4.4

2021-09-25 18:07:16 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Partially Sequential

N-Value : 6400 and Time of execution : 20.35

2021-09-25 18:07:17 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Partially Sequential

N-Value : 12800 and Time of execution : 90.5

2021-09-25 18:07:19 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Partially Sequential

N-Value : 25600 and Time of execution : 362.5

****

2021-09-25 18:07:27 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Reversed

N-Value : 50 and Time of execution : 0.0

2021-09-25 18:07:27 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Reversed

N-Value : 100 and Time of execution : 0.05

2021-09-25 18:07:27 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Reversed

N-Value : 200 and Time of execution : 0.1

2021-09-25 18:07:27 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Reversed

N-Value : 400 and Time of execution : 0.35

2021-09-25 18:07:27 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Reversed

N-Value : 800 and Time of execution : 1.4

2021-09-25 18:07:27 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Reversed

N-Value : 1600 and Time of execution : 6.0

2021-09-25 18:07:27 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Reversed

N-Value : 3200 and Time of execution : 25.5

2021-09-25 18:07:28 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Reversed

N-Value : 6400 and Time of execution : 90.05

2021-09-25 18:07:30 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

Order Type : Reversed

N-Value : 12800 and Time of execution : 360.15

2021-09-25 18:07:38 INFO Benchmark_Timer - Begin run: Benchmark_Timer Test with 20 runs

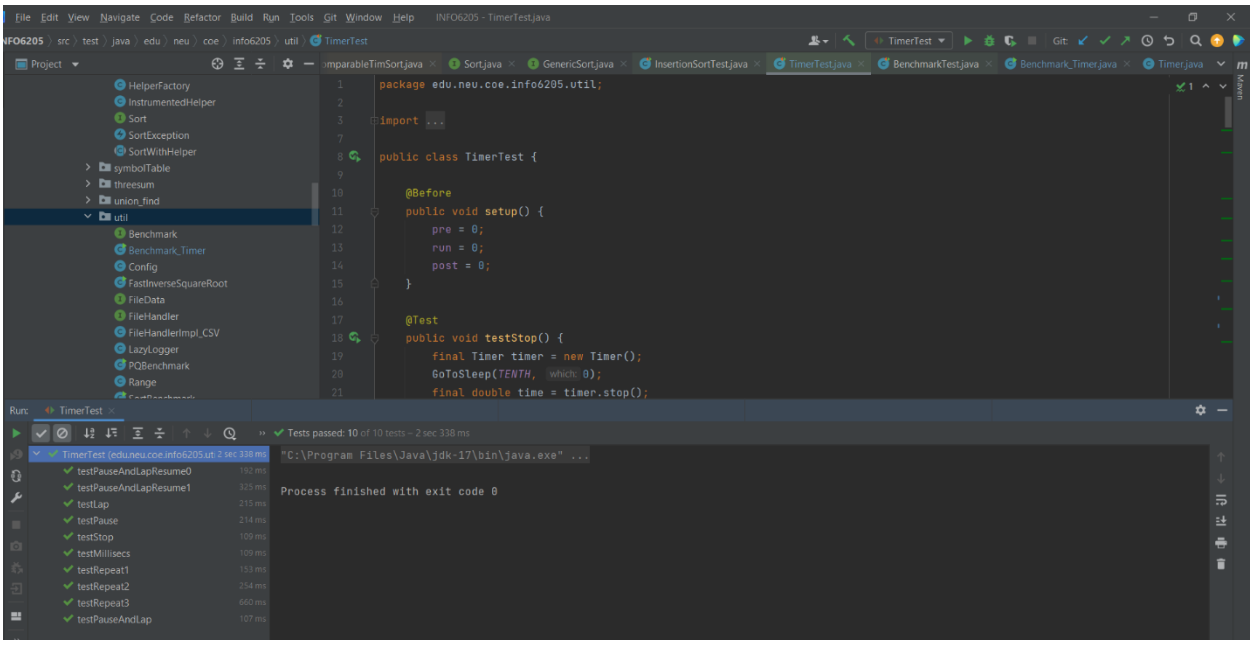
Order Type : Reversed

N-Value : 25600 and Time of execution : 1488.95

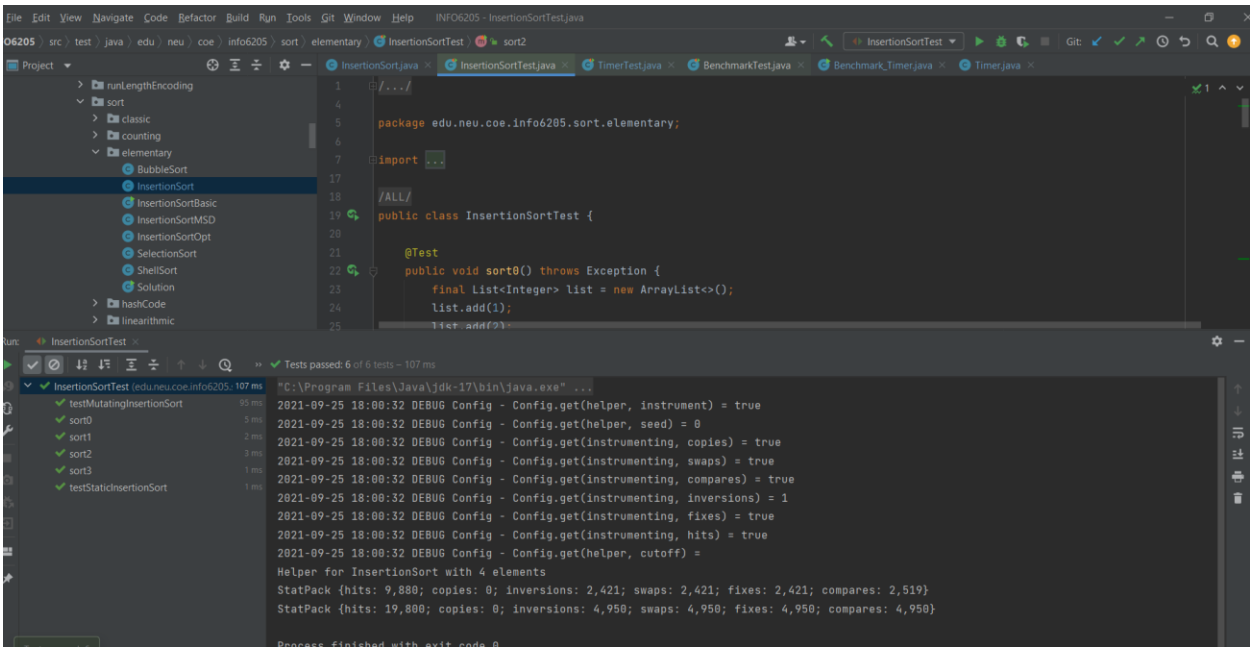
Test Case Results:

All test cases simulation ran successfully.

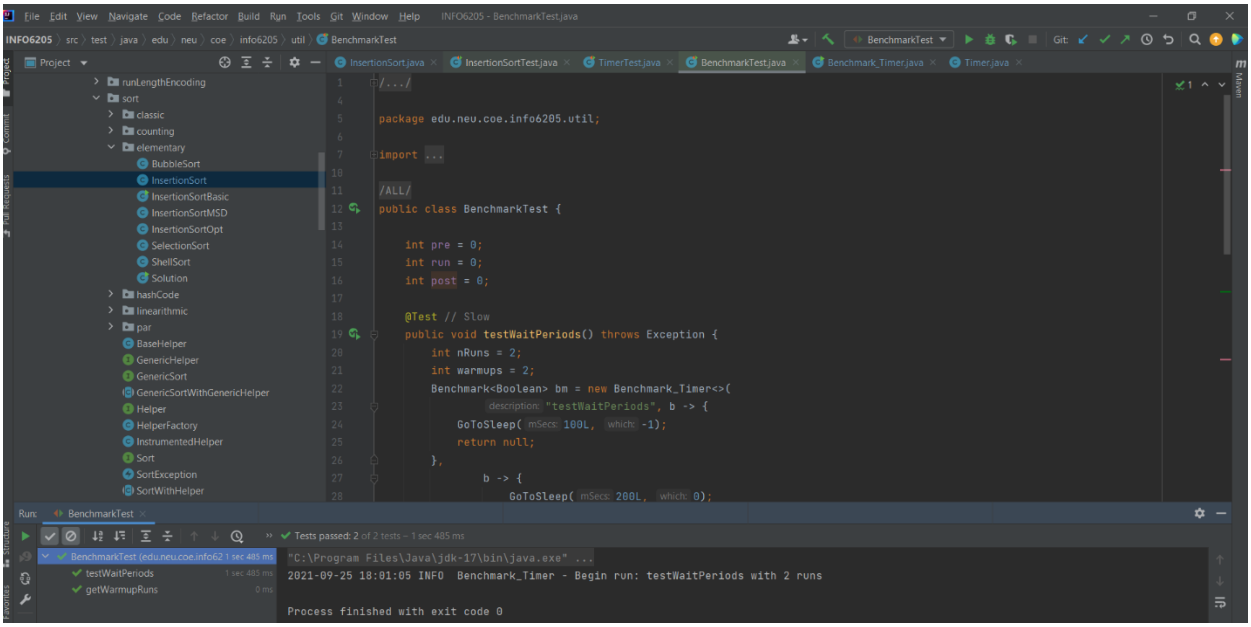
TimerTest.java



InsertionSortTest.java



BenchmarkTest.java



Deduction:

After reviewing the following experiment by running benchmark test, it can be concluded that Insertion sort has:

- **Best-case scenario** when input array is sorted. It runs in **O(n)** time, and it compares elements without swapping anyone one of them.
- **Average-case scenario** when input array is randomly ordered or is partially ordered. Then it runs in **O(n^2)** time complexity.
- **Worst-case scenario** occurs when array is sorted in the reverse order. In this case, insertion sort takes **O(n^2)** time complexity because of increased number of comparison and number of swaps required to sort the array.

Hence, running time order for different types of sorted arrays can be as follows:

Completely Ordered < Partially Ordered < Randomly Ordered < Reversely Ordered

Practical Evidence:

Following table depict the gradual increase of execution time as value of N increases. Above inferred deduction can be observed in the below table as well.

N-Value	Ordered (in ms)	Randomly Ordered (in ms)	Partially Ordered (in ms)	Reverse Ordered (in ms)
50	0.0	0.05	0.0	0.0
100	0.0	0.05	0.0	0.05
200	0.0	0.5	0.0	0.1
400	0.0	0.25	0.05	0.35
800	0.0	0.65	0.25	1.4
1600	0.0	2.75	1.4	6.0

3200	0.05	10.55	4.4	25.5
6400	0.1	37.5	20.35	90.05
12800	0.05	151.9	90.5	360.15
25600	0.05	666.75	362.5	1488.95

Charts:

The same relationship can be depicted using the multi-line chart plotted below.