Anjali Sajeevan (001563277)

Program Structures & Algorithms

Spring 2021

Assignment No. 2

• Task

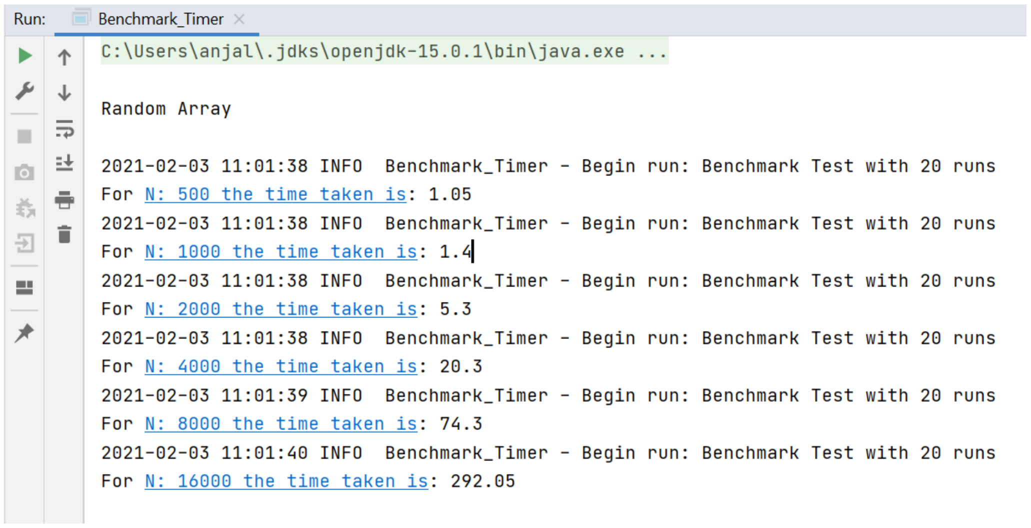
(Part 1) You are to implement three methods of a class called *Timer*. Please see the skeleton class that I created in the repository. *Timer* is invoked from a class called *Benchmark_Timer*which implements the *Benchmark* interface. The APIs of these class are as follows:

(Part 2) Implement *InsertionSort* (in the *InsertionSort* class) by simply looking up the insertion code used by *Arrays.sort.* You should use the *helper.swap* method although you could also just copy that from the same source code. You should of course run the unit tests in *InsertionSortTest*.

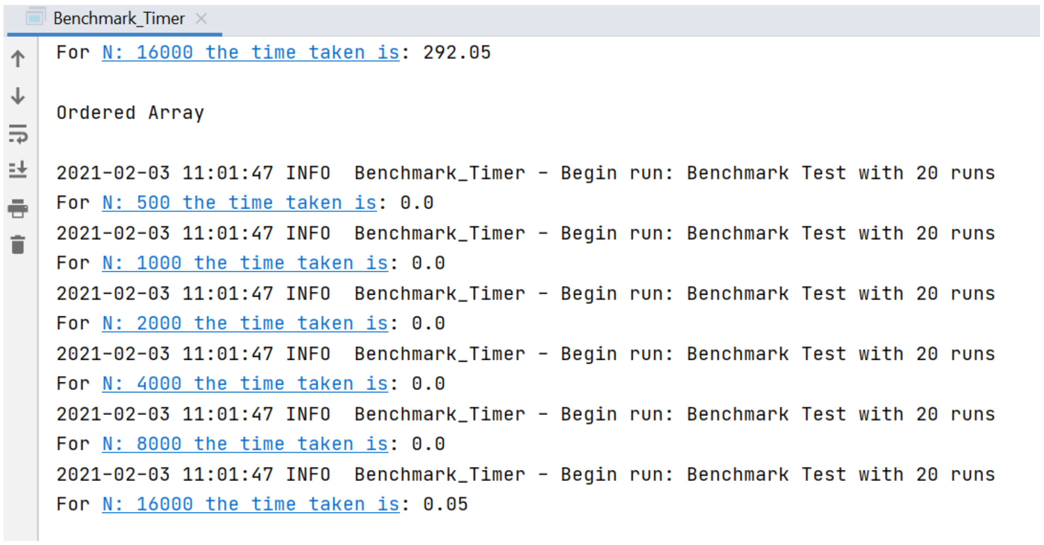
(Part 3) Implement a main program (or you could do it via your own unit tests) to actually run the following benchmarks: measure the running times of this sort, using four different initial array ordering situations: random, ordered, partially ordered and reverse-ordered. I suggest that your arrays to be sorted are of type *Integer*. Use the doubling method for choosing *n* and test for at least five values of *n.* Draw any conclusions from your observations regarding the order of growth.

• Solution

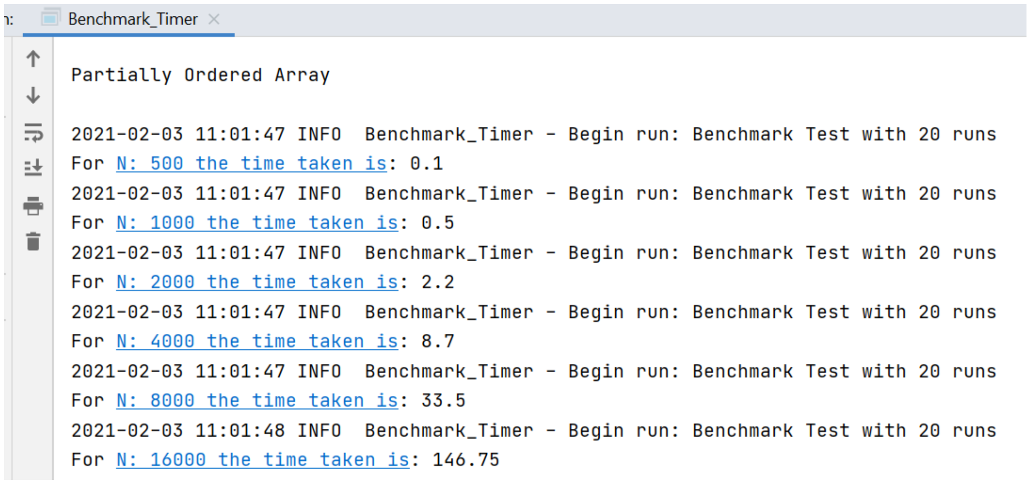
The snapshots of the result obtained for different values of n on a random array, sorted array, partially sorted array and reverse array for 20 runs are given below:



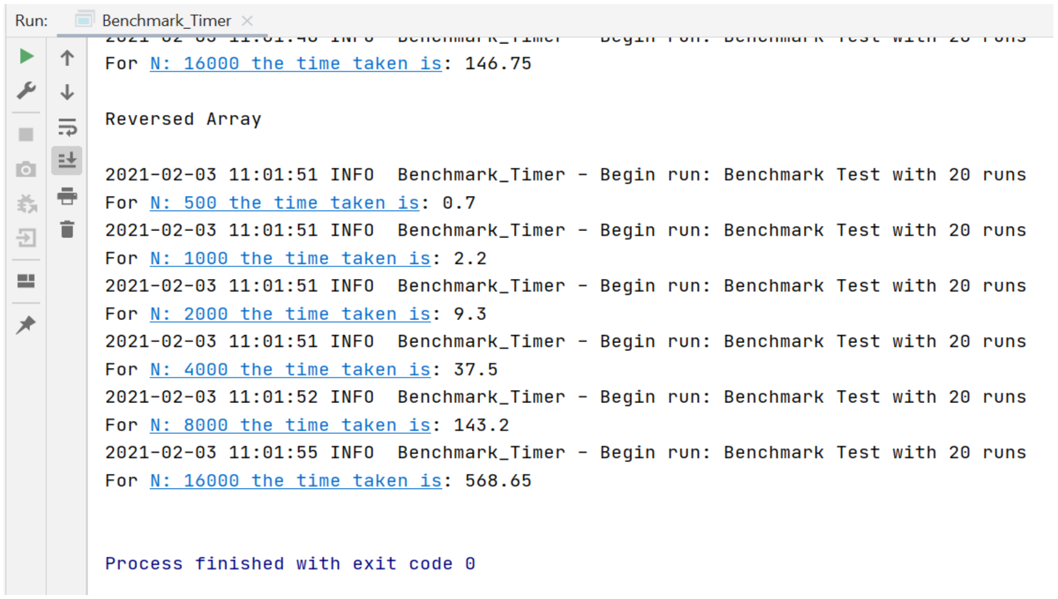
```
C:\Users\anjal\.jdks\openjdk-15.0.1\bin\java.exe ...

Random Array

2021-02-03 11:01:38 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 500 the time taken is: 1.05
2021-02-03 11:01:38 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 1000 the time taken is: 1.4
2021-02-03 11:01:38 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 2000 the time taken is: 5.3
2021-02-03 11:01:38 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 4000 the time taken is: 20.3
2021-02-03 11:01:39 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 8000 the time taken is: 74.3
2021-02-03 11:01:40 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 16000 the time taken is: 292.05
```



```
For N: 16000 the time taken is: 292.05

Ordered Array

2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 500 the time taken is: 0.0
2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 1000 the time taken is: 0.0
2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 2000 the time taken is: 0.0
2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 4000 the time taken is: 0.0
2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 8000 the time taken is: 0.0
2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 16000 the time taken is: 0.05
```



```
Partially Ordered Array

2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 500 the time taken is: 0.1
2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 1000 the time taken is: 0.5
2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 2000 the time taken is: 2.2
2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 4000 the time taken is: 8.7
2021-02-03 11:01:47 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 8000 the time taken is: 33.5
2021-02-03 11:01:48 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 16000 the time taken is: 146.75
```



```
For N: 16000 the time taken is: 146.75

Reversed Array

2021-02-03 11:01:51 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 500 the time taken is: 0.7
2021-02-03 11:01:51 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 1000 the time taken is: 2.2
2021-02-03 11:01:51 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 2000 the time taken is: 9.3
2021-02-03 11:01:51 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 4000 the time taken is: 37.5
2021-02-03 11:01:52 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 8000 the time taken is: 143.2
2021-02-03 11:01:55 INFO  Benchmark_Timer - Begin run: Benchmark Test with 20 runs
For N: 16000 the time taken is: 568.65


Process finished with exit code 0
```

- ## Relationship Conclusion:

It can be concluded from the results and the graph obtained that as the value of n increases, the time increases, where n is the size of the array and t is the run time for the insertion sort.

- The best-case scenario is when that array is sorted. Even when the value of n increases the least amount of time is taken. In a sorted array there will be no moves per stage and the number of comparisons is going to be 1 per stage hence 1+1+……+1 = (n-1) $\in O(n)$.

- The worst-case scenario is when the array is in reverse order. Even for a small value of n, the time taken is always higher for the reverse ordered array. In reverse order there will be i comparisons and moves per stage. Where i =1,2…n-1. Hence,

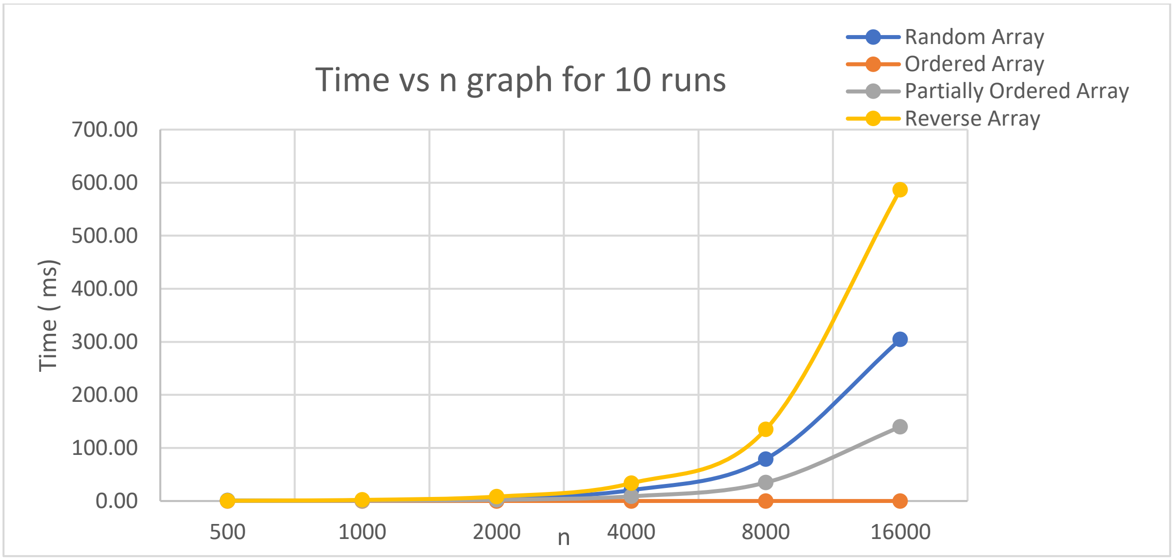
  $$=1+2\ldots+(n-1) = \frac{n(n-1)}{2} = \frac{n^2-n}{2} \in O(n^2).$$

- The time taken for partially sorted array is slightly better than the random ordered array. They both are better than the worst-case scenario but not as good as the best-case scenario.

- ## Evidence to support the conclusion and graphical representation:

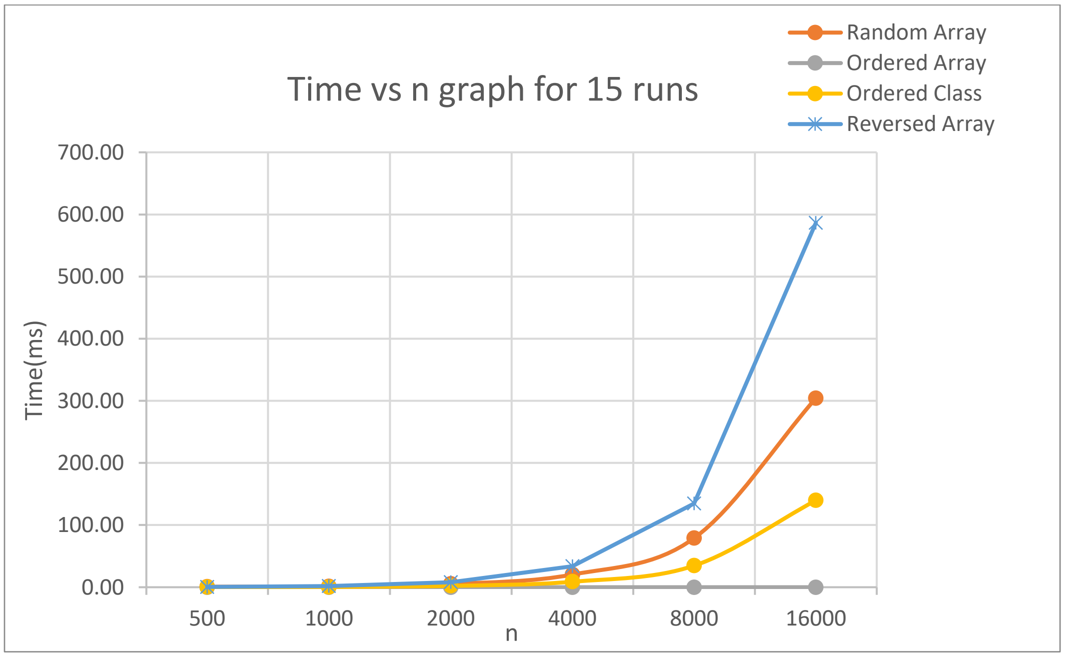
- For 10 runs

n	**Random Array**	**Ordered Array**	**Partially Ordered Array**	**Reversed Array**
500	1.00	0.00	0.1	0.5
1000	1.10	0.00	0.5	2.1
2000	5.00	0.00	2.1	8.6
4000	19.30	0.00	8.7	33.6
8000	73.20	0.00	35.6	133
16000	298.50	0.00	141.9	536.3



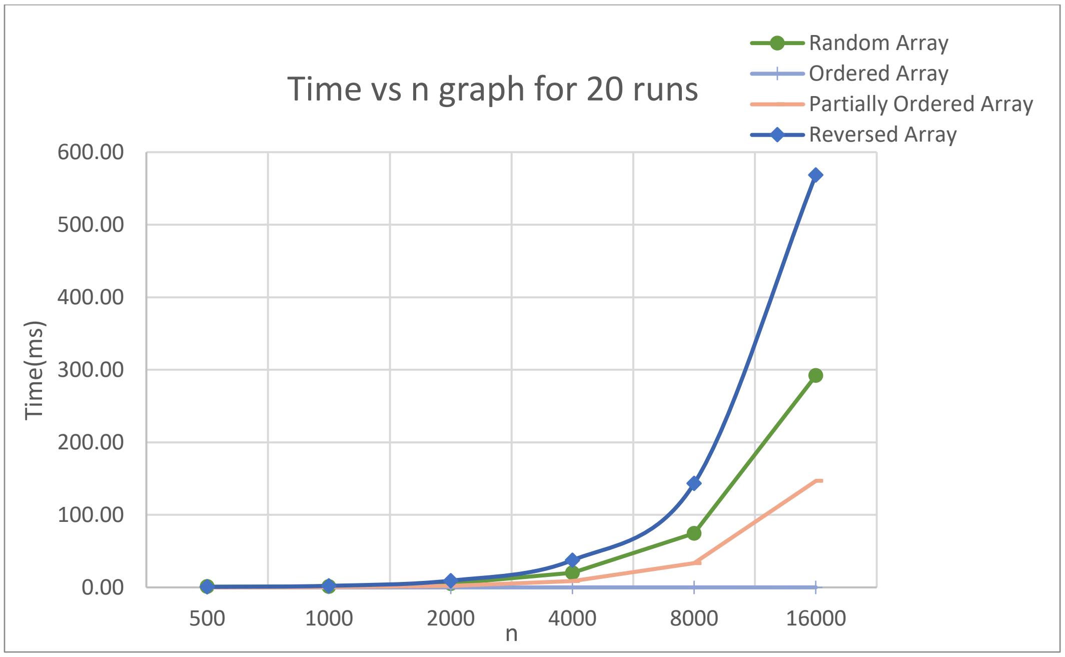
- For 15 runs

n	Random Array	Ordered Array	Partially Ordered Array	Reversed Array
500	0.80	0.00	0.13	0.53
1000	1.27	0.00	0.53	2.07
2000	5.27	0.00	2.13	8.27
4000	20.40	0.00	8.80	33.53
8000	79.00	0.00	34.73	134.93
16000	304.33	0.00	139.73	586.47



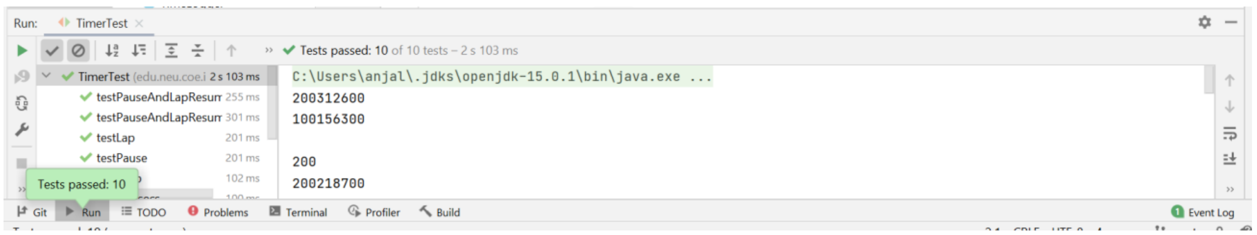
- For 20 runs

n	Random Array	Ordered Array	Partially Ordered Array	Reversed Array
500	1.05	0.00	0.10	0.70
1000	1.40	0.00	0.50	2.20
2000	5.30	0.00	2.20	9.30
4000	20.30	0.00	8.70	37.50
8000	74.30	0.00	33.50	143.20
16000	292.05	0.05	146.75	568.65



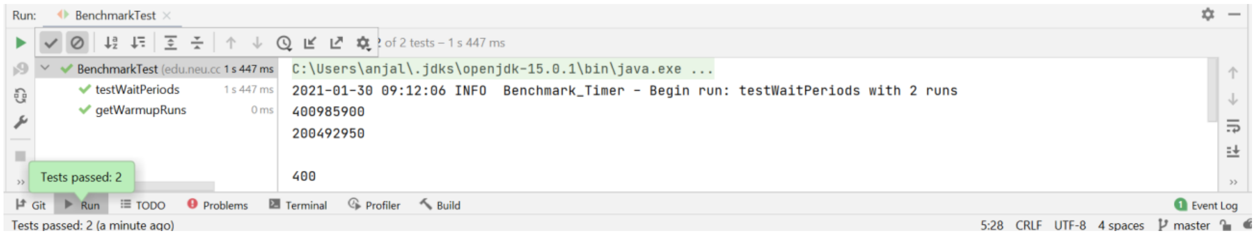
- **Unit tests result:**

1. Timer Test – Test cases result



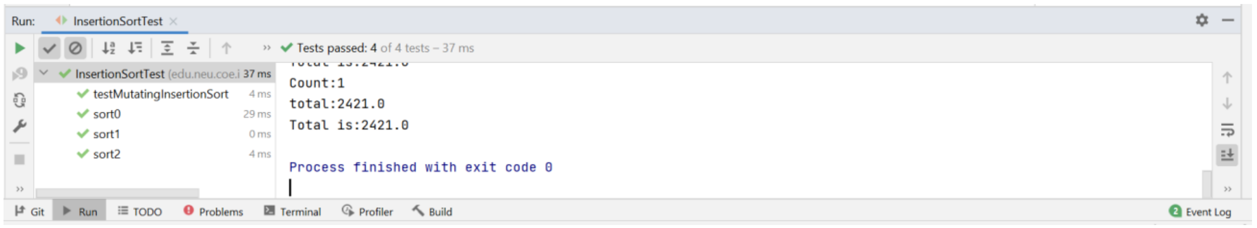
```
Tests passed: 10 of 10 tests – 2 s 103 ms
TimerTest (edu.neu.coe.i 2 s 103 ms
  testPauseAndLapResum 255 ms
  testPauseAndLapResum 301 ms
  testLap 201 ms
  testPause 201 ms
  102 ms
Tests passed: 10

C:\Users\anjal\.jdks\openjdk-15.0.1\bin\java.exe ...
200312600
100156300

200
200218700
```

2.BenchmarkTest – Test cases result


Run: BenchmarkTest
of 2 tests – 1 s 447 ms
BenchmarkTest (edu.neu.cc 1 s 447 ms
testWaitPeriods 1 s 447 ms
getWarmupRuns 0 ms
Tests passed: 2
Git Run TODO Problems Terminal Profiler Build Event Log
Tests passed: 2 (a minute ago) 5:28 CRLF UTF-8 4 spaces master

```
C:\Users\anjal\.jdks\openjdk-15.0.1\bin\java.exe ...
2021-01-30 09:12:06 INFO  Benchmark_Timer - Begin run: testWaitPeriods with 2 runs
400985900
200492950

400
```

2. Insertion Sort Test – Test cases result.


Run: InsertionSortTest
Tests passed: 4 of 4 tests – 37 ms
InsertionSortTest (edu.neu.coe.i 37 ms
testMutatingInsertionSort 4 ms
sort0 29 ms
sort1 0 ms
sort2 4 ms
Git Run TODO Problems Terminal Profiler Build Event Log

```
Count:1
total:2421.0
Total is:2421.0

Process finished with exit code 0
```