Program Structures and Algorithms
Spring 2023(SEC –1)

NAME: Foram Kamani
NUID: 002732551

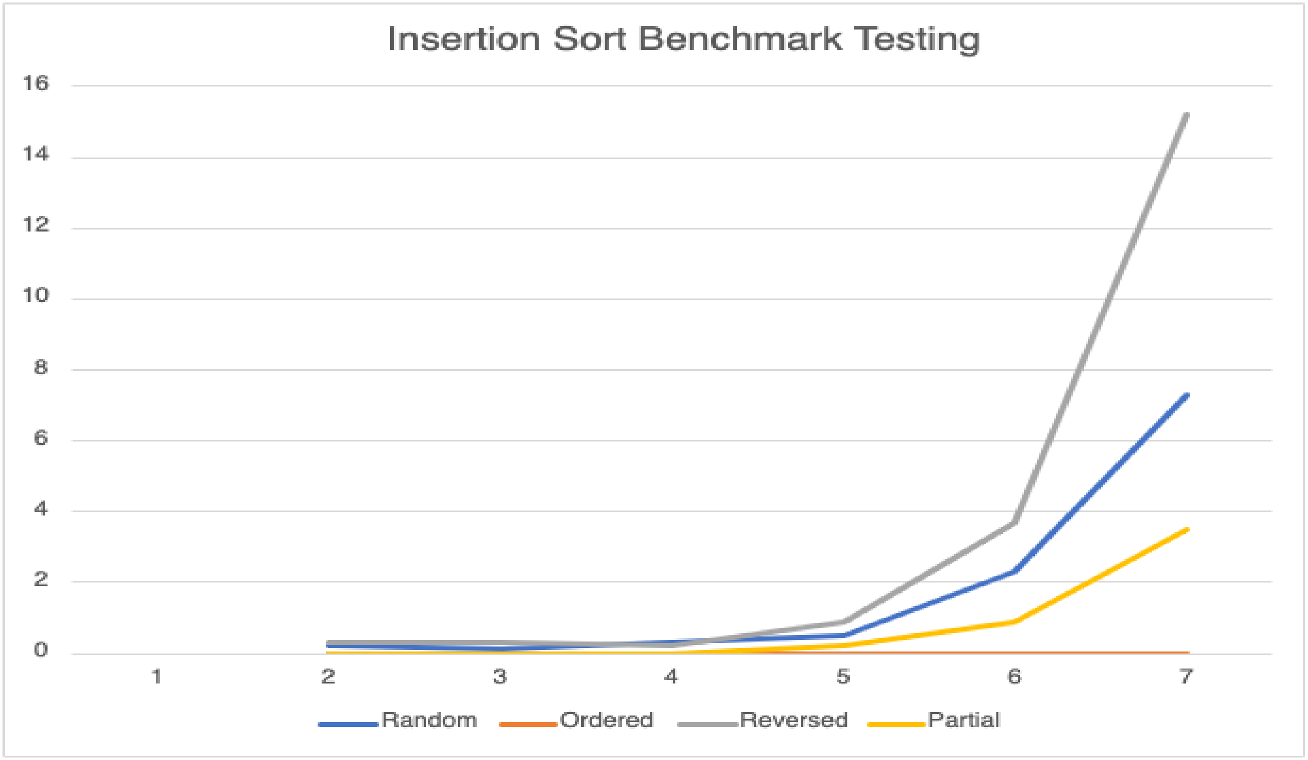
Task:

1. Implement three (3) methods (*repeat*, *getClock*, and *toMillisecs*) of a class called *Timer*. Please see the skeleton class that I created in the repository. *The timer* is invoked from a class called *Benchmark_Timer* which implements the *Benchmark* interface.
2. Implement *InsertionSort* (in the *InsertionSort* class) by simply looking up the insertion code used by *Arrays. sort.* If you have the *instrument = true* setting in *test/resources/config.ini*, then you will need to use the *helper* methods for comparing and swapping (so that they properly count the number of swaps/compares). The easiest is to use the *helper.swapStableConditional* method, continuing if it returns true, otherwise breaking the loop. Alternatively, if you are not using instruments, then you can write (or copy) your own compare/swap code. Either way, you must run the unit tests in *InsertionSortTest*.
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.

Relationship Conclusion:

N	Random	Ordered	Reversed	Partial
100	0.2	0	0.3	0
200	0.1	0	0.3	0
400	0.3	0	0.2	0
800	0.5	0	0.9	0.2
1600	2.3	0	3.7	0.9
3200	7.3	0	15.2	3.5

I conclude the following relationship between the length of the input(n) and the mean time taken for the Insertion Sort after running the main method in the Benchmark output class multiple times using the doubling method to test for different values of input arrays from array length 100 to array length 3200 for arrays of 4 different types: ordered arrays, partially ordered arrays, randomly ordered arrays and reversely ordered arrays.

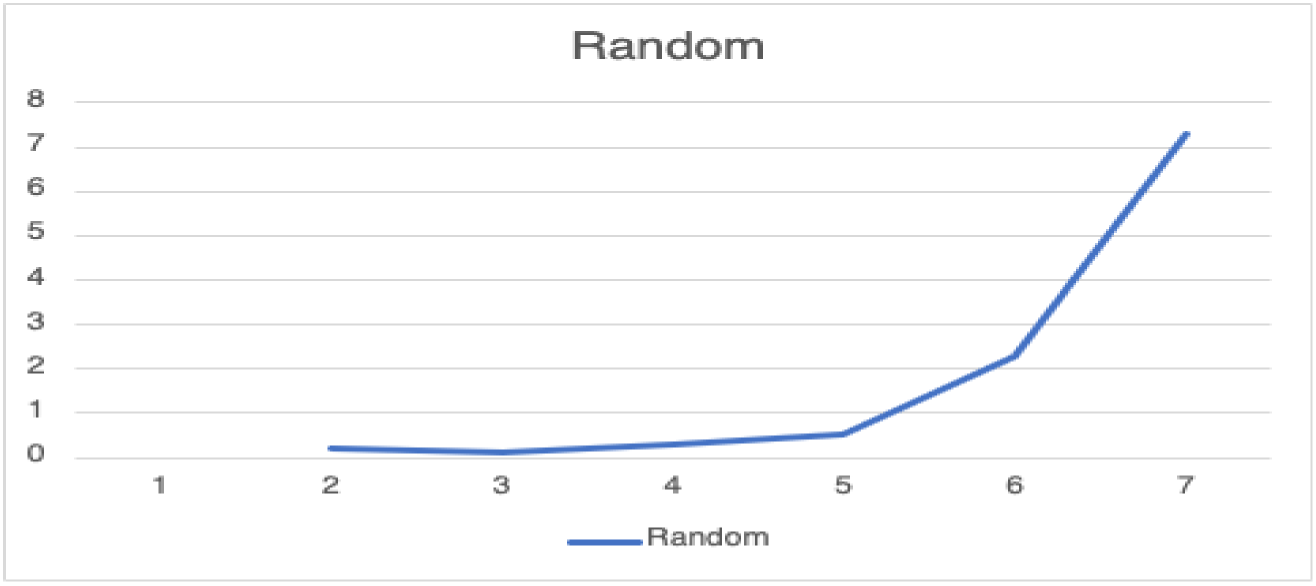


The graph above shows that insertion sort takes the most time when the array is reversely sorted because it must perform swapping for every subsequent element in the array, which is the worst-case scenario for an insertion sort algorithm. It also takes the least time when the array is already sorted. We can provide the following order of increase in time:

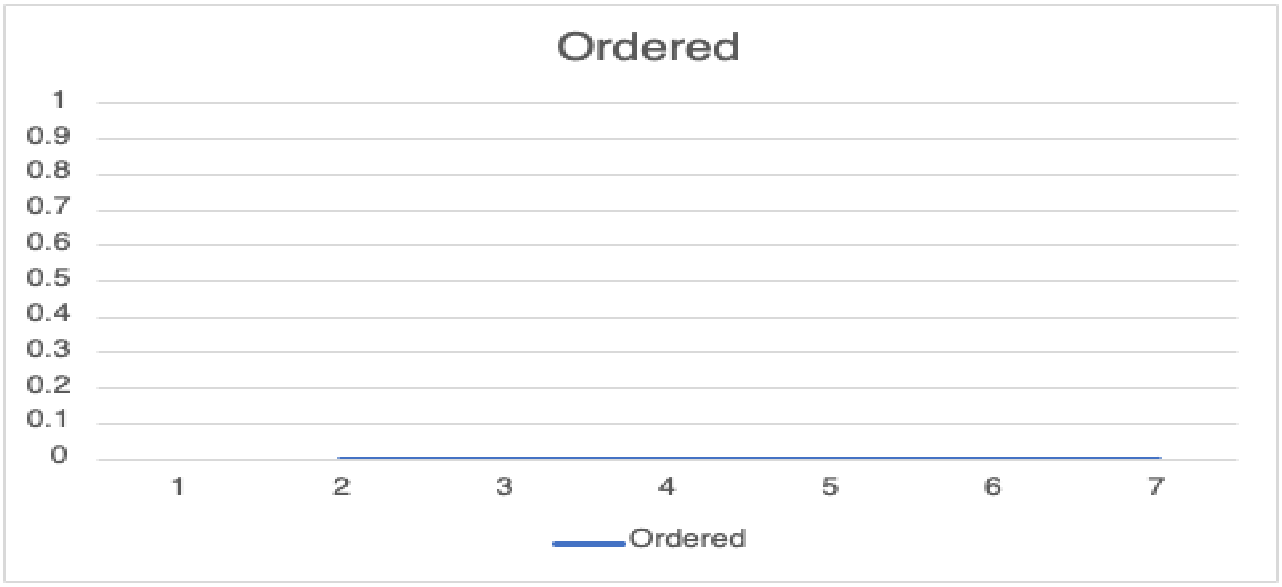
Ordered Array< Partially Ordered < Randomly Ordered < Reversely Ordered

Graphical Representation:
The graphical representations shown below are for mean times for each type of array (ordered, partially ordered, reversely ordered, randomly ordered)

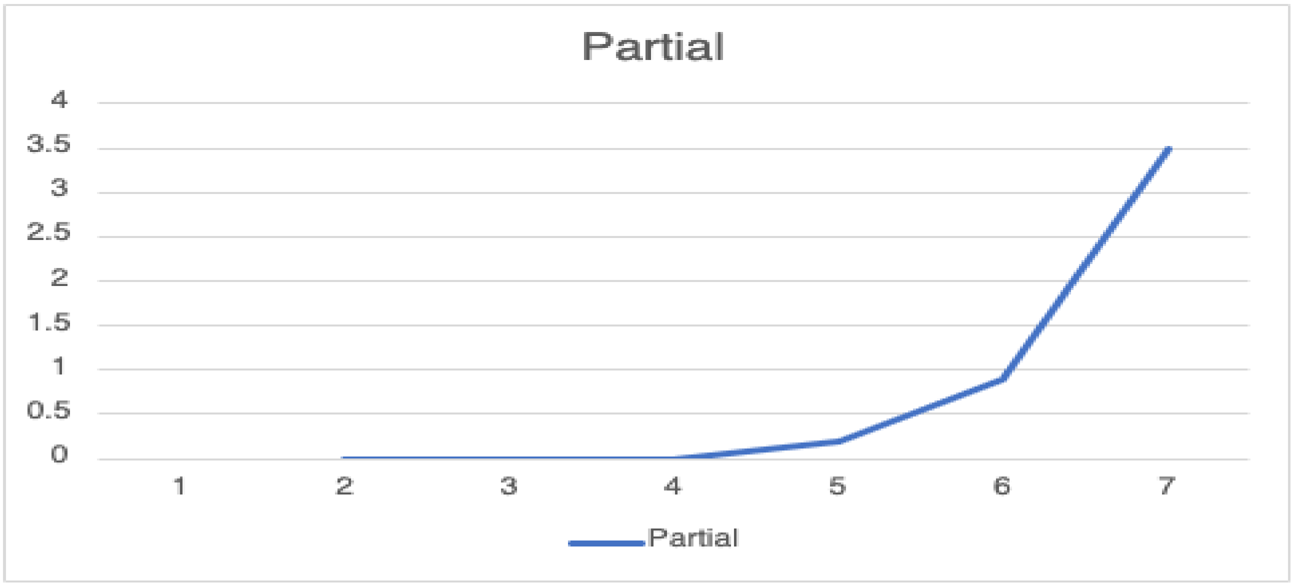
PART 1: Random



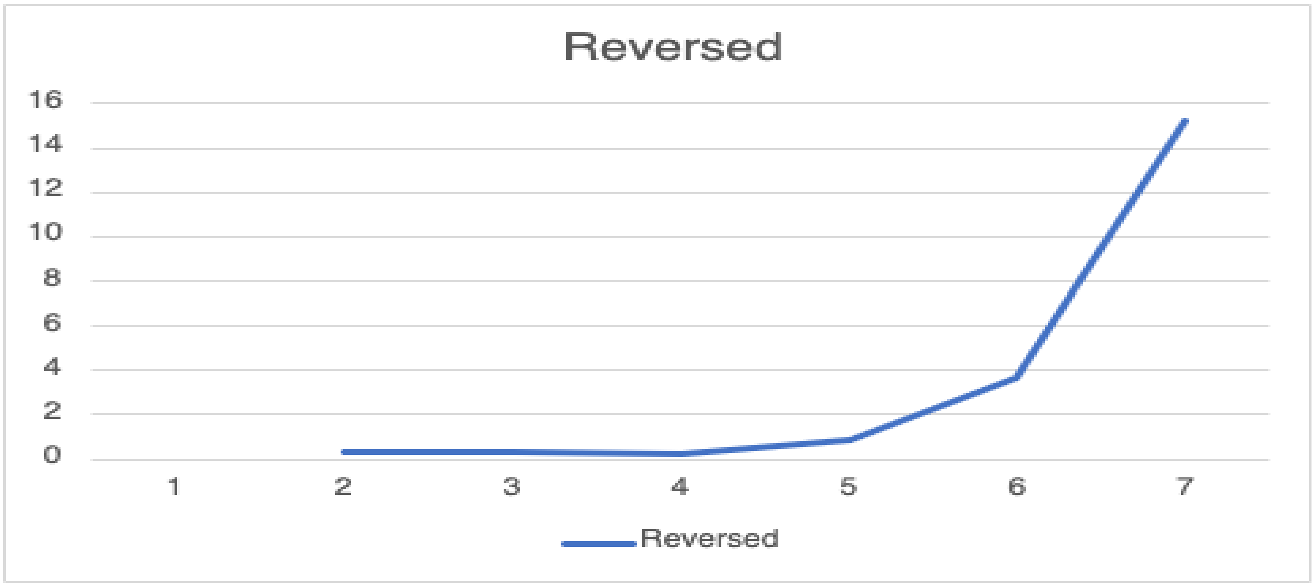
Part II: Ordered



Part III: Partially-ordered

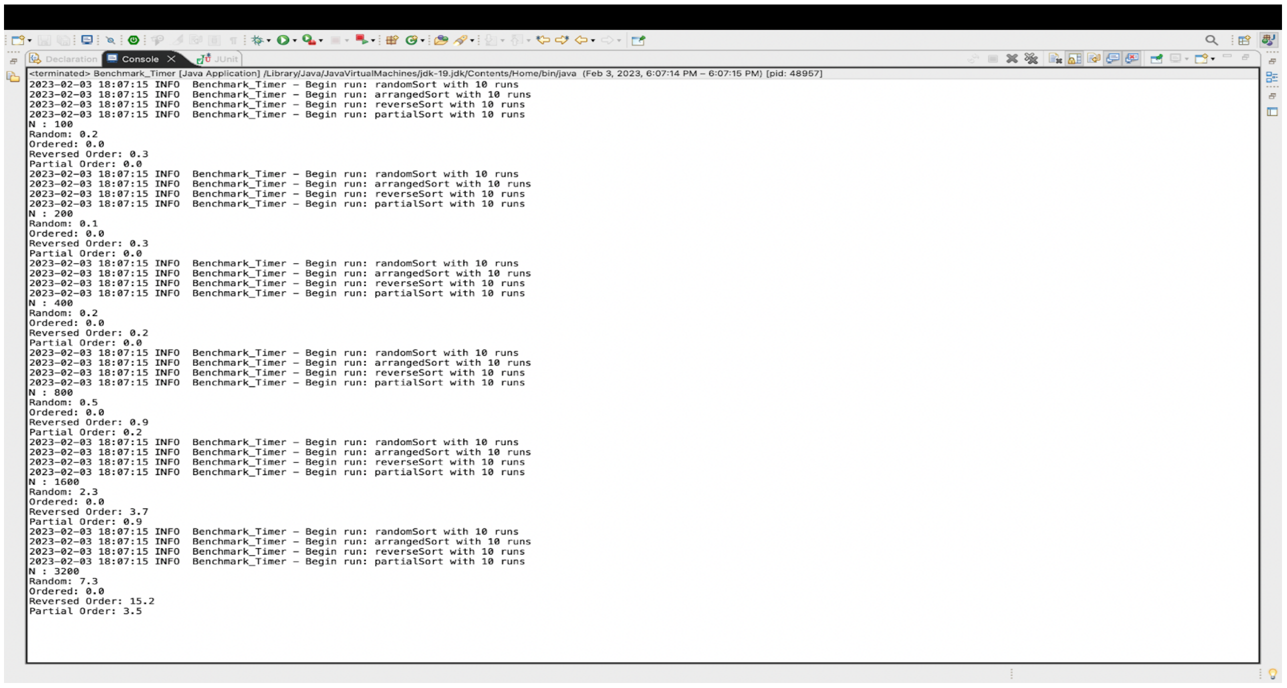


Part IV: Reverse-ordered



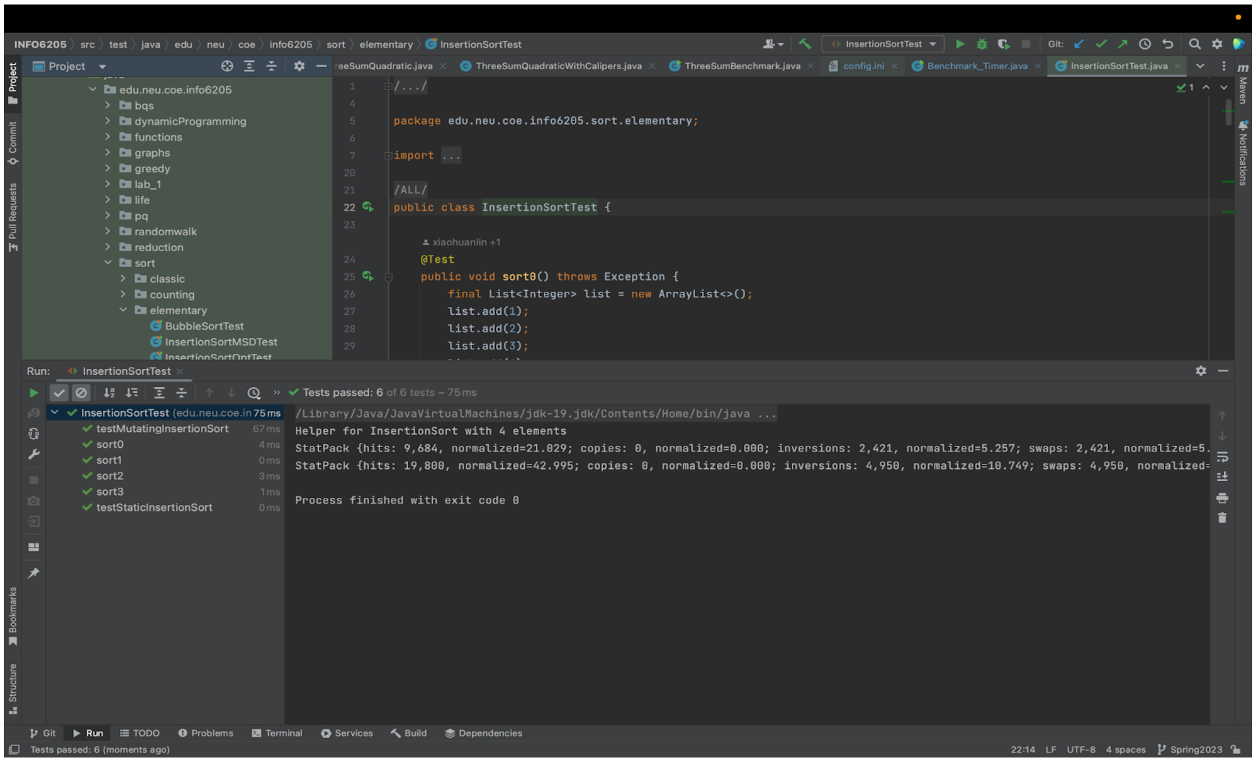
Evidence to support Conclusion:

In order to test different types of arrays (ordered, partially ordered, randomly ordered, reverse ordered) of varying lengths (n) starting from 100 going up to 3200, I have created a class with a main function to handle the different inputs.



```
<terminated> Benchmark_Timer [Java Application] /Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java  (Feb 3, 2023, 6:07:14 PM – 6:07:15 PM) [pid: 48957]
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: randomSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: arrangedSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: reverseSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: partialSort with 10 runs
N : 100
Random: 0.2
Ordered: 0.0
Reversed Order: 0.3
Partial Order: 0.0
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: randomSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: arrangedSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: reverseSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: partialSort with 10 runs
N : 200
Random: 0.1
Ordered: 0.0
Reversed Order: 0.3
Partial Order: 0.0
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: randomSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: arrangedSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: reverseSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: partialSort with 10 runs
N : 400
Random: 0.2
Ordered: 0.0
Reversed Order: 0.2
Partial Order: 0.0
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: randomSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: arrangedSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: reverseSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: partialSort with 10 runs
N : 800
Random: 0.5
Ordered: 0.0
Reversed Order: 0.9
Partial Order: 0.2
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: randomSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: arrangedSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: reverseSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: partialSort with 10 runs
N : 1600
Random: 2.3
Ordered: 0.0
Reversed Order: 3.7
Partial Order: 0.9
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: randomSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: arrangedSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: reverseSort with 10 runs
2023-02-03 18:07:15 INFO  Benchmark_Timer - Begin run: partialSort with 10 runs
N : 3200
Random: 7.3
Ordered: 0.0
Reversed Order: 15.2
Partial Order: 3.5
```

Unit Test Screenshots:



INFO6205 › src › test › java › edu › neu › coe › info6205 › sort › elementary › InsertionSortTest

```java
/.../

package edu.neu.coe.info6205.sort.elementary;

import ...

/ALL/
public class InsertionSortTest {

    xiaohuanlin +1
    @Test
    public void sort0() throws Exception {
        final List<Integer> list = new ArrayList<>();
        list.add(1);
        list.add(2);
        list.add(3);
```

Run: InsertionSortTest

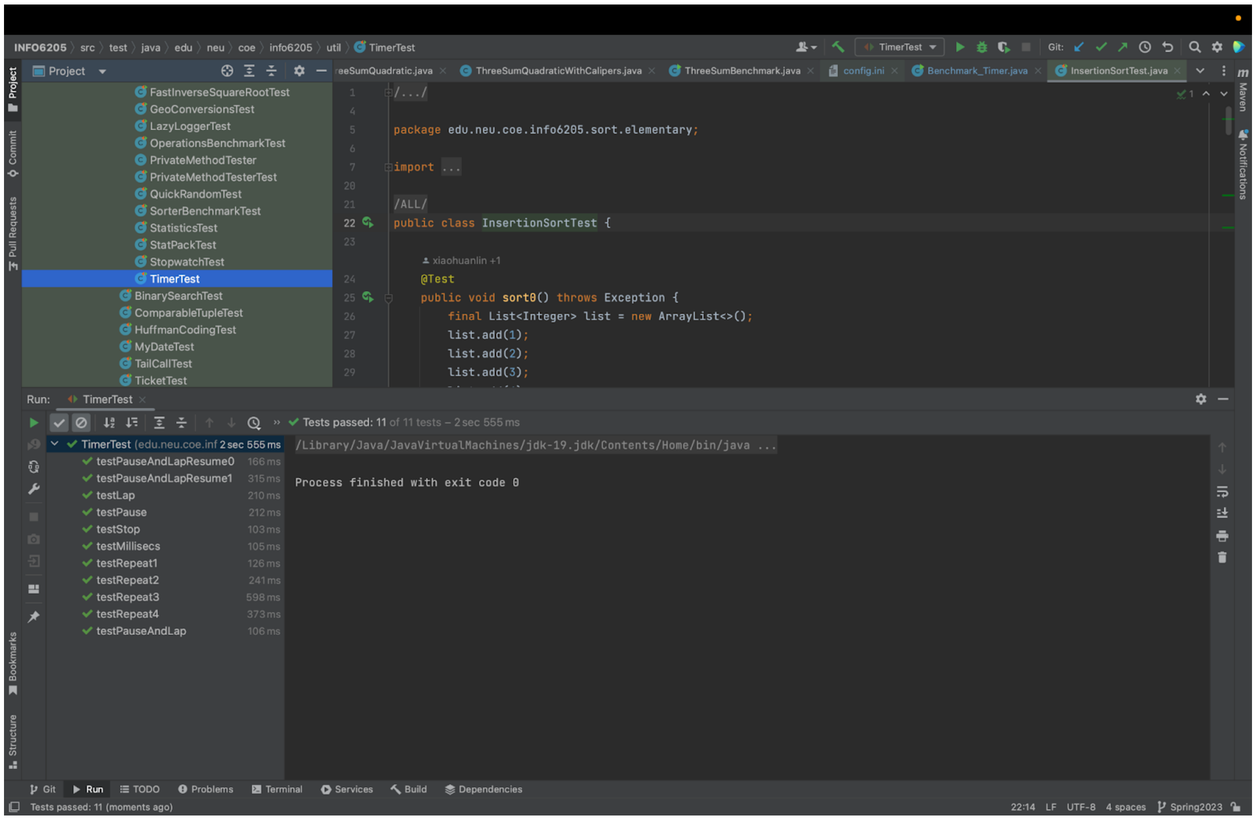
Tests passed: 6 of 6 tests – 75 ms

```
InsertionSortTest (edu.neu.coe.in 75 ms
  testMutatingInsertionSort   67 ms
  sort0                        4 ms
  sort1                        0 ms
  sort2                        3 ms
  sort3                        1 ms
  testStaticInsertionSort      0 ms
```

```
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java ...
Helper for InsertionSort with 4 elements
StatPack {hits: 9,684, normalized=21.029; copies: 0, normalized=0.000; inversions: 2,421, normalized=5.257; swaps: 2,421, normalized=5.
StatPack {hits: 19,800, normalized=42.995; copies: 0, normalized=0.000; inversions: 4,950, normalized=10.749; swaps: 4,950, normalized=

Process finished with exit code 0
```

Tests passed: 6 (moments ago)



INFO6205 › src › test › java › edu › neu › coe › info6205 › util › TimerTest

```java
/.../

package edu.neu.coe.info6205.sort.elementary;

import ...

/ALL/
public class InsertionSortTest {

    xiaohuanlin +1
    @Test
    public void sort0() throws Exception {
        final List<Integer> list = new ArrayList<>();
        list.add(1);
        list.add(2);
        list.add(3);
```

Run: TimerTest

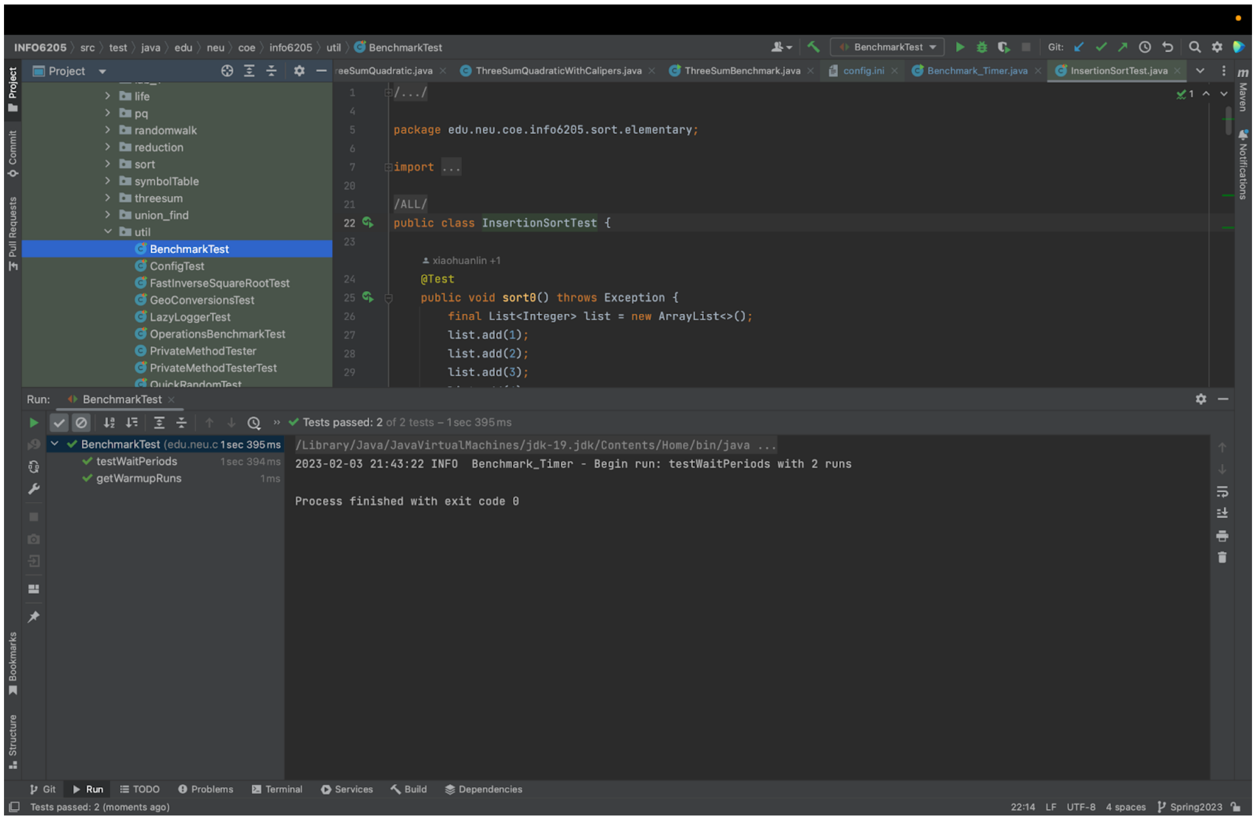
Tests passed: 11 of 11 tests – 2 sec 555 ms

```
TimerTest (edu.neu.coe.inf 2 sec 555 ms
  testPauseAndLapResume0   166 ms
  testPauseAndLapResume1   315 ms
  testLap                  210 ms
  testPause                212 ms
  testStop                 103 ms
  testMillisecs            105 ms
  testRepeat1              126 ms
  testRepeat2              241 ms
  testRepeat3              598 ms
  testRepeat4              373 ms
  testPauseAndLap          106 ms
```

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

Process finished with exit code 0
```

Tests passed: 11 (moments ago)



INFO6205 〉 src 〉 test 〉 java 〉 edu 〉 neu 〉 coe 〉 info6205 〉 util 〉 BenchmarkTest

```
/.../

package edu.neu.coe.info6205.sort.elementary;

import ...

/ALL/
public class InsertionSortTest {

    xiaohuanlin +1
    @Test
    public void sort0() throws Exception {
        final List<Integer> list = new ArrayList<>();
        list.add(1);
        list.add(2);
        list.add(3);
```

Run: BenchmarkTest

Tests passed: 2 of 2 tests – 1 sec 395 ms

- BenchmarkTest (edu.neu.c 1 sec 395 ms
  - testWaitPeriods 1 sec 394 ms
  - getWarmupRuns 1 ms

```
/Library/Java/JavaVirtualMachines/jdk-19.jdk/Contents/Home/bin/java ...
2023-02-03 21:43:22 INFO  Benchmark_Timer - Begin run: testWaitPeriods with 2 runs

Process finished with exit code 0
```

Tests passed: 2 (moments ago)